Theses and Dissertations

2013

Qualitative and Quantitative Solution Diversity in Heuristic-Search and Case-Based Planning

Alexandra Coman
Lehigh University

Qualitative and Quantitative Solution Diversity in Heuristic-Search and Case-Based Planning

by

Alexandra Coman

A Dissertation

Presented to the Graduate Committee

of Lehigh University

in Candidacy for the Degree of

Doctor of Philosophy

in

Computer Science

Lehigh University

May 2013

DISSERTATION SIGNATURE SHEET

Approved and recommended for acceptance as a dissertation in partial fulfillment of the requirements for the degree of Doctor of Philosophy.

_____

Date

_____

Dissertation Director

_____

Accepted Date

Committee Members:

_____

Dr. Héctor Muñoz-Avila (Chair)

_____

Dr. Edwin Kay

_____

Dr. Jeff Heflin

_____

Dr. Padraig O'Seaghdha

Acknowledgments

Table of Contents

List of Figures

Abstract

Planning is a branch of Artificial Intelligence (AI) concerned with projecting courses of actions for executing tasks and reaching goals. AI Planning helps increase the autonomy of artificially intelligent agents and decrease the cognitive load burdening human planners working in challenging domains, such as the Mars exploration projects. Approaches to AI planning include first-principles heuristic search planning and case-based planning. The former conducts a heuristic-guided search in the solution space, while the latter generates new solutions by adapting solutions to previously-solved problems.

The ability to generate not just one solution, but a set of meaningfully diverse solutions to each planning problem helps cater to a wider variety of user preferences and needs (which it may be difficult or even unfeasible to acquire and/or represent in their entirety), produce viable alternative courses of action to fall back on in case of failure, counter varied threats in intrusion detection, render computer games more compelling, and provide representative samples of the vast search spaces of planning problems.

This work describes a general framework for generating diverse sets of solutions (i.e. courses of action) to planning problems. The general diversity-aware planning algorithm consists of iteratively generating solutions using a composite candidate-solution evaluation criterion taking into account both how promising the

candidate solutions appear in their own right and on how likely they are to increase the overall diversity of the final set of solutions. This estimate of diversity is based on **distance metrics**, i.e. measures of the dissimilarity between two solutions. Distance metrics can be quantitative or qualitative.

Quantitative distance measures are domain-independent. They require minimum knowledge engineering, but may not reflect dissimilarities that are truly meaningful.

Qualitative distance metrics are domain-specific and reflect, based on the domain knowledge encoded within them, the kind of meaningful dissimilarities that might be identified by a person familiar with the domain.

Based on the general framework for diversity-aware planning, three domain-independent planning algorithms have been implemented and are described and evaluated herein. DivFF is a diverse heuristic search planner for deterministic planning domains (i.e. domains for which the assumption is made that any action can only have one possible outcome). DivCBP is a diverse case-based planner, also for deterministic planning domains. DivNDP is a heuristic search planner for nondeterministic planning domains (i.e. domains the descriptions of which include actions with multiple possible outcomes).

The experimental evaluation of the three algorithms is conducted on a computer game domain, chosen for its challenging characteristics, which include nondeterminism and dynamism. The generated courses of action are run in the

2

game in order to ascertain whether they affect the game environment in diverse ways. This constitutes the test of their genuine diversity, which cannot be evaluated accurately based solely on their low-level structure.

It is shown that all proposed planning systems successfully generate sets of diverse solutions using varied criteria for assessing solution dissimilarity. Qualitatively-diverse solution sets are demonstrated to constantly produce more diverse effects in the game environment than quantitatively-diverse solution sets.

A comparison between the two planning systems for deterministic domains, DivCBP and DivFF, reveals the former to be more successful at consistently generating diverse sets of solutions. The reasons for this are investigated, thus contributing to the literature of comparative studies of first-principles and case-based planning approaches.

Finally, an application of diversity in planning is showcased: simulating personality-trait variation in computer game characters. Sets of diverse solutions to both deterministic and nondeterministic planning problems are shown to successfully create diverse character behavior in the evaluation environment.

1   Introduction

Pervasive science fiction, futurology, and the computer science research community have taught us to expect our world to be permeated by an emerging artificially intelligent population, which, according to our own (diverse) dispositions, needs, and fears, we may envision pragmatically as aids to a comfortable and enjoyable life or dramatically as artificial friends, spouses, perhaps formidable foes. We are watching this fauna come to artificial life in a surreal, disconnected manner, its limbs and minds developed slowly in separate labs, an eye here, an arm there, somewhere else the ability to distinguish overlapping voices. Sprites residing in online environments already spend their time reading our minds, and nudging us towards this or that product on e-commerce websites. They have relatives who learn, discern, compare, grade homework, translate written and spoken text (even providing the right speech inflections), drive vehicles (on roads, in races and war zones, and on extraterrestrial ground), and piece together narratives. Unsurprisingly, they also plan.

It is inevitable that the Artificial Intelligence species must and will be diverse in purpose, form, abilities, and behavior: our own diversity makes it so. When diversity characterizes a particular artificially intelligent agent, it can contribute to its behaving intelligently (by making use of diverse problem-solving capabilities), appearing intelligent (by pursuing varied goals in varied ways), and, if not

necessarily appearing human, at least appealing to humans (by surprising, intriguing, enchanting).

Planning, i.e. putting together courses of actions for reaching given goals, is an essential human cognitive activity, to which we resort whenever faced with a task of at least moderate complexity, requiring, and allowing time for coming up with, more than a basic reflex response.

But why do we require synthetic entities to plan as well?

Some artificial agents have been endowed with planning abilities in order to increase their autonomy, so that they can make themselves more useful (e.g. rescue robots) or more interesting and compelling to interact with (e.g. computer game characters).

Other automated planners (such as travel planners) create plans which they do not act out themselves, but present to human users, thus assisting these users in navigating large (and, possibly, unfamiliar) solution spaces.

Sometimes, synthetic planners plan collaboratively with human experts (in "mixed-initiative"), thus relieving the human experts of some of the cognitive load (Myers *et al.*, 2002).

There are numerous approaches to Artificial Intelligence planning, and there are various ways in which to define a planning problem that is to be solved by an artificially intelligent system, based on what information characterizing a real world (or game world) issue in need of attention is encoded into the problem

description. Herein, two types of planning techniques are used to address two categories of planning problems.

In its simplest form, a planning problem specification consists of an initial state reflecting a problem situation (e.g. our friend is unhappy) and a goal state in which issues have been fixed or objectives have been reached (e.g. our friend is very happy). Domain description information (such as the available actions that can be used as the building blocks of solution plans) is provided separately, in ways specific to each planning technique.

The two categories of planning problems addressed herein are deterministic and nondeterministic planning problems.

In deterministic planning problems, it is assumed that any action will, with absolute certainty, have a specific set of outcomes (e.g. offering flowers is assumed to always result in both the recipient's joy and the jealousy of the recipient's significant other).

Unrealistic though this assumption may sound, it is not to be immediately dismissed: it was instrumental in developing solid domain-independent planning techniques, and solutions constructed under these assumptions will often work reasonably well (Orkin, 2003).

The solution of a deterministic planning problem is a *plan*: a sequence of actions which, through their predetermined outcomes, are expected to gradually change the state of the world from the initial state to a goal state.

A plan for cheering up a sad friend might consist of the sequence of actions: call friend (action 1), visit friend (action 2), offer flowers to friend (action 3). The intermediary states assumed to be brought about by actions 1 and 2 are (friend less sad) and (friend happy), respectively; while action 3 achieves the goal of the friend being very happy.

Nondeterministic planning problems may assign multiple possible outcomes to any action, thus more accurately reflecting real-world problem situations (e.g. offering flowers might result in either (1) joy or (2) an allergic sneeze and greater misery, or (3) it might be prevented altogether by unplanned-for external events).

Solutions can no longer be sequences of actions, as particular intermediary states are not guaranteed to hold. Hence, the solutions to nondeterministic planning problems are *policies:* sets of state-action pairs indicating what action should be executed in each state assumed possible.

A prudent human planner might approach things similarly by following the policy of reacting to the flower recipient's joy (state 1) with an invitation to dinner (action 1), and to an allergic reaction to the flowers (state 2) by offering an apology and an antihistamine (action 2).

The planning techniques used herein are first-principles heuristic search planning and case-based planning.

Heuristic search planners approach planning as a search, in the set of possible states of the world or in the set of possible partial plans, of a path from an initial

state/empty plan to a state in which the goal has been fulfilled/a complete plan which achieves the goals. Of course, such a search can be strenuous and inefficient if conducted blindly, so it is guided through informed guesses and clever filters: heuristics.

First-principles, or generative, planning generates each new solution from scratch, without relying on information gathered in any previous planning sessions. This, of course, is not how humans tend to plan: we often partially reuse solutions put together in previous problem-solving processes, conducted by ourselves or by others, when solving new problems (e.g. in the absence of the peonies she usually uses for a centerpiece, a florist might use chrysanthemums instead, while keeping the rest of her signature flower arrangement unchanged).

Case-based planning emulates this problem-solving approach based on analogy: solutions to new problems are created by adapting solutions to similar, previously-solved problems.

Herein, diverse planning abilities are integrated into systems conducting heuristic search planning for deterministic and nondeterministic planning problems as well as case-based planning for deterministic planning problems: hence, three diverse planning systems are proposed, differing in terms of underlying planning techniques and basic information required for planning, but subsumed by the same general diverse planning framework. The next subchapters explain what exactly is

meant herein by diverse planning, and briefly introduce this general diverse planning framework.

1.1   Defining Diversity

Diversity, variety, diversification and other related notions come up often in Artificial Intelligence literature, in reference to objectives and techniques within various degrees of similarity to the ones pursued herein.

Sometimes diversification means navigating the search space in a manner giving preference to diverse candidate solutions, so as to increase the chances of finding the optimal solution (Shell, Hernandez Rubio, and Quiroga Barro, 1994). In certain cases, the pursuit of diversity is restricted to a specific problem-solving domain, e.g. generating diverse intrusion-detection plans (Boddy *et al.*, 2005). Another possibility is for agents to be given the freedom and means to vary the goals they are seeking to achieve (Molineaux, Klenk, and Aha, 2010). Sometimes diversity is in itself the objective and specifically pursued, at other times it is merely a means to another end (e.g. efficiency, optimality, or security) and/or achieved as a side-effect of other considerations.

Herein, the specific objective is to develop domain-independent techniques for generating *sets of diverse solutions* to planning problems. Domain independence means that the algorithms will, when provided with an appropriate domain description, be able to generate solutions for problems in any application domain

(e.g. games, travel, cooking, etc.), that is, the algorithms are not tailored specifically to the characteristics of any particular domain. As the proposed techniques encompass both deterministic and nondeterministic planning, the diverse solutions can be either diverse plans or diverse policies.

Two different plans in the cheering-up domain might be a) call friend, visit friend, offer bouquet of **chrysanthemums** to friend, and b) call friend, visit friend, offer bouquet of **peonies** to friend.

Two different policies could contain the following state-action subsets, reflecting different personality traits and attitudes to friends: a) state *flowers wilted*: action *replace flowers*, (b) state *flowers wilted*: action *offer flowers*.

Why are diverse sets of solutions to planning problems useful and necessary? Diverse plans/policies can embody varied strategies and approaches to solving a problem, reflecting different priorities (such as caution versus willingness to explore in an adversarial game setting), thus catering to variation in circumstances, preferences and needs. This is particularly useful in the many situations in which it is prohibitively costly or unfeasible to specify all these preferences, circumstances, and needs as part of the problem description (in fact, they may not even be fully known at planning time).

Diverse solutions generated for human use can provide their users with alternatives sampling a larger area of the solution space (which, given the vast

search spaces often associating with planning, it is not feasible to browse exhaustively).

In mixed-initiative planning situations, by proposing a set of diverse partial plans satisfying a subgoal specified by the human planner, the artificially intelligent planner can make its human collaborator aware of possibilities that s/he may have failed to take into account (a likely scenario, given the immense cognitive load and psychological pressure burdening human planners working in information-heavy and high-risk domains, such as the Mars exploration projects, Bresina *et al.*, 2005).

When the diverse solutions are enacted by an artificial autonomous agent, they can make that agent not only better equipped to handle varied situations, but also more engaging to interact with: in computer game environments, diverse plans and policies can be used to model non-player characters exhibiting varied behavior, adding to the realistic atmosphere and enjoyment factor of the gaming experience.

1.2   Alternative Approaches to Creating Diversity

I briefly sketch several alternative approaches to generating diverse sets of solutions (in Planning and other branches of Artificial Intelligence), with which the one used herein will be contrasted (this list is not exhaustive: a more extensive overview of related work is provided in Chapter 9).

Diversity by Randomization is an accessible, knowledge-light approach to generating diverse solutions. It consists of randomizing certain decisions made

during the solution generation process. This means that no measure of diversity is used at solution generation time and, furthermore, it is not necessary to define comparison metrics (which measure how dissimilar solutions are to one another). While random diversity may be relatively easy to achieve (assuming random planning choices do not prevent a solution from being found), the value of this approach is diminished by the fact that one is generally looking not for just any set of different solutions, but for solutions which differ in a specified way.

Random diversity is often used as baseline against more targeted approaches (Smyth and McClave, 2001; Coman and Muñoz-Avila, 2011a).

Diversity by Elimination. Another approach to diversity is first generating a set of solutions, then eliminating a subset of them which are assessed as being too similar to others in the generated set (as described by Srivastava *et al*., 2007, as "a naïve approach"): this time, a measure of similarity/dissimilarity, however basic, needs to be defined. Still, diversity is not enforced while generating the set of solutions, but afterwards.

Indirect Diversity. Often, diversity comes about as a side-effect of techniques used for other purposes. In planning for computer games, diverse character behavior has been created (even serendipitously!) by endowing characters with goals or needs and with the ability to pursue their fulfillment (Orkin, 2003; Paul *et al*, 2010).

1.3 The Thesis: Iterative Diverse Solution Generation Based on Distance Metrics

My thesis is that sets of diverse solutions to planning problems can be generated, in a way that enables meaningful diversity, through an iterative approach which enforces diversity considerations specifically at solution generation time, and is based on *solution distance metrics*: measures of the dissimilarity between two solutions.

In order to generate solutions which, in addition to solving the problem, add to the diversity of the final solution set, candidate solutions are evaluated using a composite criterion (Equation 1) which balances estimated solution-set diversity with the estimated adequacy of the candidate solution. Equation 1 will be explained in more detail after introducing necessary preliminaries.

$$EvalCrit(\pi, \Pi) = \alpha SolAdequacy(\pi) + (1 - \alpha)RelDiv(\pi, \Pi) \qquad (1)$$

The general framework for this approach to generating diverse solutions specifies how to modify a regular non-diverse planner (be it heuristic-search-based or case-based) so as to obtain a diversity-aware planner.

Let us call the non-diverse planner *PL*. *PL* uses a criterion or set of criteria to evaluate and select candidate solutions during the planning process. The evaluation criteria vary from planning technique to planning technique, and even from planner

to planner (as will later be explained, estimated goal distance is a typical evaluation criterion in heuristic search planning, while similarity between the case problem and the new problem is typically used in case-based planning), and will hereinafter be referred to as the ***solution adequacy*** criterion.

Diverse solution generation is conducted in the following manner. First, a solution is generated using *PL*. Then, additional solutions are generated using a modified version of *PL* which assesses candidate solutions based on the composite evaluation criterion (Equation 1).

In Equation 1, π is a candidate partial solution to a planning problem, Π is a set of previously-generated solutions to the same problem, *SolAdequacy* is the solution adequacy of π, as computed by *PL*, *RelDiv*(π, Π) is a measure of the relative diversity between π and Π, i.e. an estimate of the diversity of the set of solutions that would be obtained by completing the current candidate solution and adding it to the set (Equation 3), while α is a parameter allowing the adjustment of the complementary weights assigned to solution adequacy and diversity (lowering α may make it difficult to find solutions; increasing α may decrease the diversity of the generated plan set).

In order for artificial planners to be able to assess whether a set of solutions is diverse, so that they can give preference to candidate solutions which appear to increase this diversity, they must be made able to achieve something human planners do naturally, though subjectively and with varying degrees of success:

14

assess just how different two solutions are from one another. Two solutions may be distinct, but are they truly meaningfully different, where what is meaningful may differ not only from domain to domain, but also from task to task? This comparison is achieved through solution distance metrics, which are of multiple types.

Quantitative solution distance metrics are defined in ways that are not specific to a particular application domain. One might, for example, count the actions which appear in either of two compared solution plans, but not the other one (Srivastava *et al*., 2007; Nguyen *et al*., 2012; Eiter *et al*., 2011; Coman and Muñoz-Avila, 2011a).

Qualitative solution distance metrics are defined based on information specific to the domain of a particular planning problem (such as knowledge of the symbolism of different types of flowers in a florist domain).

Each category has its own strengths and weaknesses: quantitative distance metrics require reduced knowledge engineering, but do not guarantee meaningful diversity; qualitative distance metrics are more knowledge-intensive, but guarantee meaningful results, as long as they are appropriately defined with regard to the domain they refer to. While the general algorithms described herein are, themselves, domain-independent, the comparison metrics they base solution differentiation on can be either quantitative or qualitative.

1.4   Contributions

I now describe the contributions that this work makes to AI planning and diversity-aware problem-solving.

1) A generalized framework (described briefly above and in detail in Chapter 3.1) for iterative diverse solution generation based on distance metrics is presented. This framework separates the solution generation algorithm from the measures used for solution comparison (distance metrics), hence it is flexible with regard to these measures. It uses a composite candidate solution evaluation criterion (Equation 1) balancing estimated solution-set diversity with solution adequacy. Implementations of the general framework can use a variety of planning techniques to solve various categories of planning problems.

2) Quantitative and qualitative distance metrics are defined (Chapter 3.2) and an experimental comparative evaluation of them is conducted. In addition, two subtypes of qualitative distance metrics, goal-achievement and inflation-based distance (Chapter 3.4) are defined and illustrated in an experimental environment.

3) This work contributes to the body of literature comparing first-principles and adaptation-based approaches to planning by presenting the first comparison between a first-principles planner and a case-based planner

from the point of view of the *diversity* of the generated solutions (Chapter 5). Previous studies have compared such systems in terms of planning efficiency and other considerations (Veloso, 1994; Gerevini and Serina, 2000; 2010; Fox *et al.*, 2006; Au, Muñoz-Avila, and Nau, 2002; van der Krogt and de Weerdt, 2005; Kuchibatla and Muñoz-Avila, 2006).

4) All diverse-solution-generation algorithms described herein are tested on a non-synthetic planning domain based a computer game. This evaluation is conducted by running the generated plans and policies in the game environment, observing the effects they have on the environment, and quantifying and analyzing the diversity of these effects. This is a significant contribution: in previous work, the diversity of generated plans was only assessed by analyzing the sets of plans themselves.

Based on the general diverse planning framework, a set of domain-independent diverse planning algorithms have been developed. These algorithms address deterministic and nondeterministic planning, and all of them are flexible with regard to the solution comparison metrics they use.

DivFF (Chapter 4.1) is a diverse heuristic search planner for solving deterministic planning problems. DivCBP (Chapter 4.2) is a diverse case-based planner, also for deterministic planning problems. DivNDP (Chapter 6) is a heuristic search planner for solving nondeterministic planning problems. DivCBP

is the first system for diverse case-based planning, while DivNDP is the first system for diverse non-probabilistic nondeterministic planning.

A possible application for solution diversity is proposed and explored: simulating personality-trait variation in non-player characters in computer games and other virtual environments (Chapter 7). In fact, all diverse planning techniques presented herein are demonstrated and tested in a computer game environment. The relationship between Artificial Intelligence and computer games is an immediately obvious and mutually beneficial one. Clever artificial agents make engaging non-player characters to populate virtual worlds with (Orkin, 2003) and, in return, game domains provide Artificial Intelligence researchers with testbeds (Ontañón *et al.*, 2010) sharing many challenging characteristics (such as nondeterminism and dynamism) with other domains of practical interest, for which they can act as risk-free simulation environments.

1. 5   Outline

The remainder of this dissertation is organized as follows. Chapter 2 introduces relevant background on deterministic and nondeterministic planning domains, first-principles heuristic search planning for deterministic and nondeterministic planning domains, and the case-based approach to planning. Definitions necessary for understanding the upcoming material are provided. Inspiration from and

connections with human approaches to planning and general problem-solving, as studied by Cognitive Science, are also addressed.

Chapter 3 introduces and motivates the Solution Diversity problem, as addressed herein, in more detail. It also presents the general framework for diverse planning which subsumes all diverse planning systems proposed in this work. The various categories of distance metrics (quantitative and qualitative, goal-achievement and inflation-based) are described, and a planning domain based on the computer game Wargus, which will be used extensively in the experimental evaluation, is introduced.

Chapter 4 is dedicated to diversity in planning for deterministic domains. First, it describes a general framework for diverse planning based on diverse heuristic search, and introduces DivFF, an implementation of it. Then, it addresses diversity in case-based planning: it begins with a differentiation between state diversity and plan diversity, necessary due to the approach to diversity previously taken in case-based reasoning; then, it presents a general algorithm for diverse case-based planning.

Chapter 5 provides a comparative study of DivFF and DivCBP, primarily from the point of view of the diversity of their generated solutions.

Chapter 6 describes DivNDP, an algorithm for generating diverse policies for nondeterministic planning problems.

In Chapter 7, computer-game-character diversity is showcased as an application of plan and policy diversity.

Chapter 8 is dedicated to the experimental evaluation of all the proposed diverse planning systems.

An overview of related work is presented in Chapter 9. It includes diversity in subfields of AI other than planning, other approaches to comparing solutions and generating diverse solutions, and other approaches to modeling computer game characters and ensuring their diverse behavior.

Chapter 10 is dedicated to the conclusion and ideas for possible future work extending various research directions explored herein.

2   Background

Planning is a branch of Artificial Intelligence concerned with generating courses of action for reaching given goals, maximizing utility functions, and performing given set of tasks.

Typically (though not exclusively), a planning problem is described in terms of an initial state (or set of states) and a goal state (or set of states). Its solution is a plan (sequence of actions) or a policy (set of state-action pairs) which, when executed, gradually transforms the initial state(s) into a goal state.

Deterministic planning (which generates plans) is based on the assumption that each action that may be part of a plan has a predefined outcome, assumed to occur with absolute certainty every time the action is executed.

Nondeterministic planning (which produces policies) allows for planning domain descriptions in which actions have multiple possible outcomes.

Planning has been conducted using a wide variety of problem-solving techniques and modeling frameworks, including: search algorithms with or without heuristic guidance, constraint-satisfaction and propositional satisfiability techniques, Markov decision processes, and model-checking techniques (Ghallab, Nau, and Traverso, 2004). Knowledge-engineering requirements range from relatively low (e.g. basic specifications of available actions, with preconditions and effects) to high, as in the case of Hierarchical Task Network planning, which, based

on an extensive description of how tasks are solved in a particular application domain, conducts an iterative deconstruction of high-level tasks to lower level ones, until a plan consisting of elementary actions has been produced.

Domain-independent planning systems are able to generate plans for any domain, as long as they are provided with an appropriate domain description.

Domain-dependent planning systems are tailored to the characteristics of particular application domains (e.g. cooking) and may take advantage of characteristics of the domain to increase the effectiveness and efficiency of planning.

In the next subchapters, I describe deterministic and nondeterministic planning domains and problems as well as two planning techniques: heuristic-search planning and case-based planning. I also introduce the heuristic-search planner FastFoward (FF) and the NDP algorithm for nondeterministic planning, which will later be modified so as to be able to produce diverse sets of solutions.

2.1   Deterministic Planning

Classical planning (Ghallab, Nau, and Traverso, 2004) is a planning paradigm based on assumptions which, by abstracting away problematic characteristics of realistic domains, have helped develop efficient domain-independent planning techniques, perhaps at the expense of immediate applicability to real-world

problems (although solutions developed under these assumptions will sometimes work reasonable well even when run in the environments in which the assumptions do not hold).

Perhaps the most notable simplification at the basis of classical planning is that of **determinism**: the assumption that any action has one predefined outcome assumed to occur with absolute certainty, hence: executing action a in state s will always result in a given state s'.

The additional assumptions of classical planning include the following:

- fully-observable environment: complete information about the state of the environment is always available;

- static environment: the state of the environment can only be changed by the agent's actions, i.e. there are no external events that can influence the current state;

- implicit time: there is no notion of action duration, transitions from one state to another are assumed to be instantaneous;

- finite set of states: the environment may only ever be in one of a finite set of states;

- offline planning: no planning or replanning needs to be conducted during plan execution, meaning that environment conditions are expected to be as assumed at planning time, and goals are expected not to change.

A classical **planner** receives as input a problem description consisting of an initial state and a goal state, and a planning domain description (a state-transition system). Its output is as solution **plan** that solves the input problem by transforming its initial state into the goal state.

Definition 1. A **deterministic planning domain** *Dom* is a triple *Dom = (S, A, γ),* where *S* is a finite set of states, *A* is a finite set of actions, and $\gamma : S \times A \rightarrow 2^S$ is the **state-transition function**, which describes the environment in which it is assumed that the plan will be executed. This environment can be in various states, and it can be influenced through actions. Actions cause the environment to change its current state to another state. If $\gamma(s, a)$ is not empty, then action *a* can be applied while the environment is in state *s*, causing the current state to change from *s* to $\gamma(s, a)$.

In practice, the description of a planning domain (to be used by a heuristic search planner such as FastForward, by Hoffmann and Nebel, 2001, which will be described in detail later on), typically consists of the following elements: a set of objects (which have specified types) assumed to exist in the world, a set of first-order literals with formal parameters of specified types (these formal parameters can be instantiated with available objects of the proper type), used to represent facts that may hold in the domain world, such as relationships between objects; and a set of operators with preconditions and effects.

Definition 2. A **state** is a set of facts represented as first-order literals (this is the *classical state representation* in the state classification of Ghallab, Nau, and

24

Traverso, 2004). For example, `(isSad(friend), have(freesias))` might be the description of a possible state of the world in which our friend is sad and we are in the possession of a bouquet of freesias.

Definition 3. An **operator** is a generalized representation of an action in a planning domain. It specifies a set of formal parameters, a set of preconditions, and a set of effects defined in terms of these formal parameters.

Definition 4. An **action** is an instantiation of an operator *o*, obtained by instantiating the formal parameters of *o* with actual objects from the domain description. For example, the action `has(friend, freesias)` could be an instantiation of the operator `has(Person, Thing)`.

Definition 5. A **precondition** of operator *o* is a fact that needs to hold in state *s* (i.e., a literal that needs to be contained in the set of literals describing *s*) in order for an action *a* instantiating operator *o* to be applicable in state *s*. The preconditions of action *a* are instantiations of the preconditions of operator *o*. For example, if `have(Thing)` is a precondition of the operator `give(Thing, Person)`, then `have(freesias)` is a precondition of the action `give(freesias, friend)`.

Definition 6. An **effect** of operator *o* specifies a way in which the current state will be modified when an action *a*, which is an instantiation of *o*, is executed. Effects are of two types: **add effects** (facts which are added to the description of state *s* in

25

order to obtain state *γ(s, a)*), and **delete effects** (facts which are removed from the description of state *s* in order to obtain the description of state *γ(s, a)*). The effects of action *a* are instantiations of the effects of operator *o*. For example `has(Person, Thing)` could be an add effect of the action `give(Thing, Person)`. A delete effect of the same action might be `not have(Thing)`.

Definition 7. An action *a* is **applicable** in state *s* if the preconditions of *a* hold in *s*: formally, if $\gamma(s, a)$ is nonempty. $A_{\text{Dom}}(s)$ is the set of all actions applicable in *s* in the domain *Dom*. If $a \in A_{\text{Dom}}(s)$, then executing *a* in *s* results in the state $\gamma(s, a)$.

Definition 8. A **deterministic planning problem** is a triple $P = (Dom, s_0, g)$, where *Dom* is a deterministic planning domain, $s_0 \in S$ is an initial state, and *g* a set of facts making up the goal.

The solution of a deterministic planning problem is a **plan**.

Definition 9. A **plan** π is a sequence of actions $(a_1,...,a_n)$, which, through their add and delete effects, repeatedly change the state of the environment, taking it from an initial state s_0, through a series of states $(s_1,\ldots,s_{n-1})$, to a final state s_n. In order for π to be a valid plan for the planning domain *Dom* and the planning problem *P*, the preconditions (as defined in the description of *Dom*) of action a_0 must hold in the initial state of *P*, and the preconditions of each action a_i must hold in the state $\gamma(s_{i-2}, a_{i-1})$. A valid plan π is a **solution** for planning problem *P* if, after executing the

sequence of actions π consists of, the final state of the environment contains all facts that make up g, the goal of P.

Types of planning which eliminate the classical planning assumptions include *nondeterministic planning* (which eliminates the determinism and, sometimes, the fully-observable-environment assumptions), conformant planning (which does not require states to be observable), temporal planning (which eliminates the implicit time assumption), partially-observable planning (which does not require the current state of the environment to be fully known), and online planning/replanning (which consist of adjusting plans during execution, in response to unexpected environment conditions and/or changing goals). An upcoming subchapter will be dedicated to nondeterministic planning techniques.

2.2   Heuristic-Search Planning

Heuristic search planners approach planning problems as search problems. They conduct planning iteratively, at each step generating and evaluating a set of partial candidate solutions, out of which one candidate is chosen for further refinement, until a valid solution has been constructed (or a failure signal has been triggered). Heuristic functions are used to evaluate and select candidate partial solutions.

Heuristics provide educated (but certainly not infallible) guesses as to the value of a candidate solution, typically in the form of an estimate of the effort required to turn the candidate solution in question into a complete, valid solution to the

planning problem. The lower the estimated effort, the more promising the candidate solution. Heuristic evaluations are often conducted based on a solution of a **relaxed problem** (a simplified version of the planning problem), assumed to provide a good enough approximation of a solution of the actual problem.

State-space search heuristic planners (such as the one used later on in this work) conduct search in a subset of the space of possible states of the world.[1] At each stage of the search, a number of candidate states are evaluated using a heuristic function. After the most promising candidate *s* is chosen, states in the neighborhood of *s* are identified and become the new candidate states. What the initial value of *s* is, and which states are in the neighborhood of *s* depends on whether a forward-search or a backward-search planner is used. In state-space planning, a typical evaluation heuristic is **goal distance**: an estimate of the length of the plan from the candidate state to a state satisfying the goal.

Forward-search planners start search from the initial state of the problem, and look for a state containing all facts specified in the goal description. They identify states in the neighborhood of *s* by applying to *s* the effects of actions applicable in *s*.

---

[1] They can be contrasted with **plan-space** heuristic search planners, which search in the space of possible partial plans.

Backward-search planners begin searching from a state consisting of all the facts in the goal description. The neighborhood of the current state s consists of the set of states S', such that, for each state $s' \in S'$, $\gamma(s', a) = s$.

In the context of forward-search state-space heuristic planning, the term **candidate solution plan** will be used to refer to a plan consisting of the sequence of actions $(a_1,\ldots,a_i, a_c)$, where the current state $s = s_i$, and $\gamma(s_i, a_c) = s_c$, where s_c is a candidate state in the neighborhood of s. In other words, a candidate solution plan is a plan leading from the initial state of the problem to one of the current candidate states. It is not to be confused with candidate solution plans in plan-space search planning, the components (and their ordering) of which are subject to modification as long as a commitment is not absolutely necessary. In state-space search, the first segment of a candidate solution plan (i.e. the sequence of all actions but the final one) has already been established as being part of the final solution plan, and the final action is the only one still subject to selection.

The term *candidate solution* is preferred herein to *candidate state*, as it allows one to refer more generally to similar constructs in various types of heuristic search planning as well as case-based planning.

Some heuristic-search-based planners are also **first-principles** (or **generative**) **planners**, i.e. they produce new solutions from scratch. An example of such a planner is FastForward (Hoffmann and Nebel, 2001), which will be described in more detail later on.

Algorithm 1: General Framework for Heuristic Search Planning

Input: *PL* - a heuristic search planner which uses the heuristic function h_{PL} to evaluate candidate solutions, and *P* - the planning problem.

Output: a solution plan π.

getCandidates(*P*, π) returns the set of candidate partial solutions to problem *P* in the neighborhood of partial solution π.

selectCandidate(*C*, h_{PL}) returns the candidate partial solution in *C* that is ranked highest based on the heuristic h_{PL}.

GeneratePlan(*PL*, *P*)

1. $\pi \leftarrow$ empty-plan/seed plan

2. Repeat

3. $C \leftarrow$ getCandidates(*P*, π)

4. $\pi \leftarrow$ selectCandidate(*C*, h_{PL})

5. Until π is a solution for *P*

6. Return π

The abstracted pseudocode above (Algorithm 1) provides a generalized sketch of how heuristic search planning (be it state-space or plan-space, forward or backward) works.

2.2.1 The FF Heuristic Search Planning System

FastForward (or *FF*, Hoffmann and Nebel, 2001) is a first-principles, heuristic, forward-search, state-space planner which uses a goal-distance planning heuristic function for candidate-solution evaluation.

The heuristic value of a candidate state is the length of the solution plan of a relaxation of the planning problem, with the candidate state as initial state. The relaxed problem is obtained by ignoring all delete effects of the actions in the planning domain. The solution plan to the relaxed problem is generated using the Graphplan planner (Blum and Furst, 1997), which conducts planning by building a structure called planning graph, from which it then extracts a solution plan.

The primary search type used by FF, Enforced Hill-Climbing (EHC), searches the state space in the following way: starting from the initial state, for each selected current state s, it conducts a breadth-first search, in the set of states that are reachable[2] from s, for the first state that is better evaluated than s (based on the heuristic described above) or fulfills the goal of the planning problem. Note that not all candidate states[3] will be evaluated, as search stops as soon as a state with a good

---

[2] Note that this search is not restricted to states in the immediate vicinity of s (states which can be obtained by applying only one action), but can be extended to states obtainable by applying multiple actions.

[3] Unlike in the more limited definition for general state-space search planning, candidate states are now all states which are reachable from s, by any number of steps.

enough evaluation is found. If there are no more candidate states to explore and the goal has not been reached, EHC is considered to have failed.

EHC also conducts preliminary action pruning using a filter called *Helpful Actions*. This filter makes use of information extracted from the Graphplan planning graph to eliminate from consideration applicable actions which look less promising. However, as this evaluation is not infallible, useful actions can be filtered out, leading to the goal being unreachable.

While efficient, EHC is not complete: it is not guaranteed that, if a solution exists, EHC will find it. When it fails to produce a solution, FF switches to a complete variant of Best-First Search (BFS), with no action pruning.

Note that while, with BFS, heuristics are still used to guide the search, in the hope of increasing efficiency, the entire search space will be explored if necessary. Hence, if a solution to the problem exists, it will, in theory, be found (in practice, prohibitive memory and time requirements may prevent this from being achieved within reasonable cost limits). The ability to fall back on BFS whenever EHC fails makes the FF planner complete.

2.3   Case-based Planning

Case-based planning (Spalazzi, 2001; Cox, Muñoz-Avila, and Bergmann, 2005) is a subbranch of case-based reasoning (Aamodt and Plaza, 1994), a "psychologically-plausible" problem-solving paradigm modeled on an approach to

fulfilling tasks employed routinely by humans: using memory and analogy-based reasoning to create solutions to new problems by making use of information from previous problem-solving sessions (Hammond, 1990; Keane, 1996; Spalazzi, 2001).

While first-principles planning systems put together solutions to a new problem from scratch, case-based planning systems identify previously-solved problems similar to the current one, then make use of stored information from these previous planning processes to produce solutions to the new problem.

Case-based planners are either **transformational-analogy planners** (as is the one used herein) or **derivational-analogy planners**.

Briefly, transformational-analogy planners modify solutions to previously-solved problems to obtain solutions for the new problem, whereas derivational-analogy planners attempt to re-conduct, in the context of the new problem, a planning process previously used for other problem-solving tasks. The information that needs to be stored for future use and adapted to the new context is, in the first case, the solution itself, and, in the second case, a description of the planning process that has produced the solution.

While, in theory, the case-based planning approach could be used to solve both deterministic and nondeterministic planning problems, work conducted so far has focused on deterministic planning.

Case-based planning systems plan using **case bases** containing information collected during previous problem-solving processes. The input these planning systems require is a **new problem** ("new" as opposed to the already-solved "old" problems in case-base cases).

Definition 10. A **case base** is a set of **cases** containing information from previous planning processes. A **case** consists of a previously-solved planning **problem** and information regarding the **solution** of the problem: the actual solution plan for transformational-analogy systems, and a derivational trace summarizing a previous planning process for derivational-analogy systems.

Definition 11. A **case-based planning problem** is typically a triple $P = (CB, s_0, g)$, where CB is a case base, $s_0 \in S$ is an initial state, and g a set of facts making up the goal. It may contain additional information, such as the reasons for the failure of previous planning attempts.

The case-based planning cycle (Spalazzi, 2001), based on the case-based reasoning problem-solving cycle, as described by Aamodt and Plaza (1994) generally consists of the following steps (although specific approaches taken by individual case-based planners vary greatly, and may include modifications of the cycle):

- **retrieval** from the case base of one or more appropriate solution cases (chosen based on criteria such as **similarity metrics** assessing the relevance of stored cases to the current problem);

- **adaptation (reuse):** adapting the retrieved case(s) as necessary, in order to create a valid solution for the new problem (adaptation can be achieved in a number of ways, including by domain-dependent or domain-independent heuristics, constraint satisfaction, merging multiple retrieved cases, and utilizing first-principles planning systems);

- **revision** of the generated plans, in case of planning or plan-execution failure;

- **retaining** the newly-generated plans in the case base, for use in future planning processes.

Case-based planners can be combined with heuristic search planners, which can be used in various stages of the case-based planning cycle.

I now describe the retrieval and adaptation stages, which are the most relevant to the work presented herein, in more detail.

Retrieval of a case or set of cases from the case base is conducted based on a **retrieval criterion** which, most commonly, is a measure of the similarity between the case problem and the new problem. A typical approach is to the rank cases based on the retrieval criterion, and then return the top k cases.

For example, in the florist domain, the similarity criterion might be the type of the event at which a flower arrangement is meant to be displayed. If the new problem requires a bouquet for a wedding, we might select $k = 3$ cases the

problems of which required flower arrangements for 1) a wedding, 2) a wedding anniversary, 3) an engagement party.

Criteria other than similarity can and have been used for retrieval: an example of such an approach is adaptation-guided retrieval (Smyth and Keane, 1998), which chooses cases based on an assessment of how easy it would be adapt their solutions to the requirements of the new problem.

Retrieval criteria vary in computational complexity, based, among others, on the complexity of the problem representation. A two-step retrieval method (Forbus, Gentner, and Law, 1995) may be used if the retrieval criterion is expensive to compute: in such cases, a more approachable criterion is used to filter out a subset of the cases; then, the expensive criterion is be applied only to the remaining cases, in order to select the final top k ones.

While, traditionally, in general case-based reasoning, retrieval is conducted based on the problem-description component of the case, in certain problem-solving domains, including diverse planning (as will be shown later on), it is necessary to take the solution into account at retrieval time as well. Two-stage retrieval might be a good approach to this issue, as applying retrieval criteria to solutions is likely to become computationally expensive in complex planning domains.

Adaptation, or reuse, can be achieved in various ways, as described by Spalazzi (2001) and Muñoz-Avila and Cox (2008).

Sometimes, no adaptation is conducted, and retrieved plans are returned as they are, to be used directly or adapted manually (because of the prohibitive difficulty of automated adaptation for particular tasks or users' distrust of its capabilities). When used, adaptation techniques can be either domain-independent or domain-specific.

Transformational analogy adapts a solution plan from a retrieved case through insertion, deletion, and/or modification of the plan's components. In the florist domain, the action of adding daffodils to a bouquet might be replaced with that of adding yellow freesias if daffodils are not available, or are disliked by the client. Generative heuristic-search planning techniques can be used at this stage.

In derivational analogy, adaptation consists of using a retrieved derivational trace to attempt to replicate previous planning processes in the context of a new problem (typically using a first-principles planning system to do so), to the extent that this possible (for example, certain planning decisions specified in the derivational trace may not be feasible in the current planning context because the required preconditions do not hold).

Outside case-based planning, the term *plan adaptation* can be seen as referring, more generally, to any planning approach which makes use of information from previous planning processes to create new plans, e.g. **plan repair** (Fox *et al*, 2006; van der Krogt and de Weerdt, 2005).

Finally, I will make a note of the fact that, while planning tasks are **synthesis tasks**, i.e. they consist of putting together a new solution (a plan), case-based reasoning is also extensively used to solve **analysis tasks**, which only require preexisting library entries to be selected, labeled, or classified. This distinction is a necessary prerequisite for understanding various discussions later on.

2.4   Nondeterministic Planning

Nondeterministic planning eliminates the classical planning assumption of determinism, allowing planning domains to include actions with multiple possible outcomes. There are two main types of nondeterministic planning: **probabilistic** and **non-probabilistic** (Ghallab, Nau, and Traverso, 2004).

In **probabilistic planning**, it is assumed that the possible outcomes of an action have been assigned probability values indicating the likelihood of their occurrence. This allows goals to be encoded as objective functions and problems to be treated as optimization tasks.

Non-probabilistic planning, which does not make these assumptions, more accurately reflects characteristics of the many domains in which it is unrealistic or impractical to assign probabilities to the outcomes of the actions, as pointed out by Ghallab, Nau and Traverso (2004).

Hereinafter, the term nondeterministic planning will be used to refer to non-probabilistic nondeterministic planning, unless otherwise noted.

The input of a nondeterministic planning system consists of a **problem description** (a set of initial states and a set of goal states) and a **domain description** (a state-transition function reflecting the fact that actions may have multiple possible outcomes). The output is **solution policy**.

Assume the following definitions (Kuter *et al.*, 2008).

Definition 12. A **nondeterministic planning domain** *Dom* is a triple *Dom* = (*S*, *A*, γ), where *S* is a finite set of states, *A* is a finite set of actions, and γ : $S \times A \rightarrow 2^S$ is the state-transition function. An action *a* is applicable in *s* if γ(*s*, *a*) is nonempty. $A_{\text{Dom}}(s)$ is the set of all actions applicable in *s* in the domain *Dom*. If $a \in A_{\text{Dom}}(s)$, then executing *a* in *s* may result in any one of the states in γ(*s*, *a*).

Definition 13. A **nondeterministic planning problem** is a triple *P* = (*Dom*, S_0, *G*), where *Dom* = (*S*, *A*, γ) is a nondeterministic planning domain, *A* is a set of actions, $S_0 \subseteq S$ is a set of initial states, and $G \subseteq S$ is a set of goal states.

Definition 14. A **policy** is a function π: $S_\pi \subseteq S \rightarrow A$, such that, for each $s \in S_\pi$, the action π(*s*) is applicable in *s*. Intuitively, a policy is a set of state-action pairs (s_i, a_i), indicating that, when s_i is identified as the current state of the environment, action a_i should be executed.

Definition 15. The **execution structure** of a policy π, Σ_π, is a directed graph representing all possible executions of π. $\Sigma_\pi = (V_\pi, E_\pi)$, where $V_\pi = S_\pi \cup \cup_{s \in S\pi}$

$\{\gamma(s, \pi(s))\}$ and $E_\pi = \{(s, s') \mid s \in S_\pi, s' \in \gamma(s, \pi(s))\}$. If there is a path in Σ_π from s to s', then s' is a π-descendant of s.

Cimatti *et al.* (2003) define three types of solutions for nondeterministic planning problems: weak, strong, and strong cyclic.

Definition 16. **Strong cyclic solutions** (the type of solutions generated herein) guarantee that any possible execution path will reach a goal in a finite number of steps, provided it exits any loop that it enters. Strong cyclic solutions are safer than **weak solutions** (which are only required to have one possible execution path which reaches a goal), and less restrictive than **strong solutions** (which must guarantee that a goal will be reached in a finite number of steps, hence not allowing loops), so that a problem which does not have strong solutions may have strong cyclic solutions.

Like Kuter *et al.* (2008) and Fu *et al.* (2011), this work addresses strong-cyclic planning for **fully-observable nondeterministic (FOND)** problems. FOND problems are defined under the assumption that the states are fully observable. Even under this assumption, the planning difficulty lies in the potential exponential growth of the search space in domains containing actions with numerous possible outcomes, all of which must be taken into account.

2.4.1 The NDP Nondeterministic Planning System

NDP (Kuter *et al*., 2008) is an algorithm for solving nondeterministic planning problems by converting nondeterministic planning domains into deterministic ones, then repeatedly calling a deterministic heuristic planner, and step-wise putting together a policy from the plans it generates (Algorithm 2).

FOND domains are converted into deterministic ones by replacing each nondeterministic action a, where a has n possible outcomes, with n deterministic actions $a_1,..,a_n$, where each a_i corresponds to one outcome of the original action a.

Let *PL* be the heuristic planner called by NDP. On the first run, if successful, *PL* produces a complete plan p_1 from an initial state to a goal state. During planning, each time a deterministic action a_i originating from a nondeterministic action a is added to the partial plan, all states corresponding to effects of a other than the intended one, called *failed effects* (Fu *et al*., 2011), are added to a set of open nodes. At each subsequent stage, until no open nodes remain, an open node is chosen arbitrarily, and *PL* attempts to generate a plan p_k from this open node to a goal state, again adding any new failed effects to the list of open nodes. After generating each of these plans p_k, all state-action pairs (s, a), where a is the nondeterministic action corresponding to the deterministic action a_i in p_k, and s is a state in which a_i is applied in p_k, are added to the partial policy. The policy is complete when no open nodes remain.

Algorithm 2: The NDP Algorithm for Solving FOND Planning Problems

Input: *Dom* - a nondeterministic planning domain, S_0 - the set of initial states of the nondeterministic planning problem, *G* - the set of goal states of the problem, and *PL* - the classical planner that NDP makes use of.

Output: π, a solution policy for the planning problem (*Dom*, S_0, *G*).

NDP(*Dom*, S_0, *G*, *PL*)

1. *Dom'* $\leftarrow$ a classical relaxation of *Dom*

2. $\pi \leftarrow \varnothing$

3. $S_0 \leftarrow S_0 \backslash G$

4. If $S_0 = \varnothing$

5.   Return π

6. Loop

7.   If $\exists s$ in S_0 s.t. $A_{\text{Dom'}}(s) = \varnothing$

8.   Return FAILURE

9. $S \leftarrow$ {all non-goal leaf states in $\Sigma_\pi(S_0)$}

10.   If $S = \varnothing$

11. $\pi \leftarrow \pi \backslash \{(s, a) \in \pi \,|\, s$ is not a π-descendant of $S_0\}$

12.   Return π

13.   arbitrarily select a state $s \in S$

14.   call *PL* on the planning problem (*Dom'*, *s*, *G*)

15.   If *PL* returns a solution plan *p* then

16. $\pi \leftarrow \pi \cup \{(s, a) \mid a_i$ is an action in p, a is the non-deterministic action corresponding to a_i, and s is the state in which a_i is applied in $p\}$

17.   Else //*PL* returns FAILURE

18.   BACKTRACK[4]

Fu *et al*. (2011) implement a variant of NDP which uses FF (Hoffmann and Nebel, 2001) and is augmented with two extensions called **state reuse** and **goal alternative**. Their version is shown to be characterized by higher planning efficiency and smaller generated policies (seen as a sign of solution quality, as pointed out by Kuter *et al*., 2008) than previous strong-cyclic planners including MBP (Cimatti *et al*., 2003) and Gamer (Kissmann and Edelkamp, 2009).

State reuse is achieved by stopping an FF search as soon as any solved state (a state which already has an associated action in the partial policy) has been reached, instead of always continuing until a goal state has been reached.

Goal alternative consists of providing open nodes with alternative goals (the alternative goal of a node representing a failed effect is the intended effect). The path to an alternative goal can be reasonably assumed to be shorter than the path to

---

[4] As described by Kuter *et al*., 2008.

an actual goal of the problem. This enhancement is intended to reduce the effort required for calculating the FF goal-distance heuristic.

Nondeterministic planners that generate strong-cyclic solutions using algorithms other than NDP include MBP (Cimatti *et al*., 2003), which uses symbolic model-checking and Binary Decision Diagrams (BDDs) for compact state representation; Gamer (Kissmann and Edelkamp, 2009), which also uses BDDs and converts nondeterministic planning problems to two-player turn-based games, and the system of Mattmüller *et al*. (2010), which uses LAO* heuristic search (Hansen and Zilberstein, 2001).

2.5 Naturally and Artificially Intelligent Planners: Cognitive Science and Automated Planning

Planning is a fundamental human cognitive activity: it is our approach to tasks ranging from minor day-to-day challenges of limited consequence to complex endeavors and the pursuit of long-term goals (Berger, Guilford, and Christensen, 1957; Hayes-Roth and Hayes-Roth, 1979; Mumford, Schultz, and Van Doorn, 2001). It is also a crucial component of communication and social interaction (Berger and DiBattista, 1993; Berger, 2007; Dillard, 2008; Miller Henningsen *et al*., 2011). In a Cognitive Science experiment on conversational tactics, it was estimated that 44% of the subjects' conscious thoughts were dedicated to the goals

they were pursuing and to planning for these goals (Waldron, 1990). Berger and DiBattista (1993) consider this to be an underestimation.

Artificial Intelligence and the study of human approaches to problem-solving (including planning), have been evolving in a mutually-influencing manner. Newell and Simon, co-authors of a seminal AI general-purpose problem-solving algorithm (Newell, Shaw, and Simon, 1959) are also the creators of a model of human problem solving (Newell and Simon, 1972). This model is based on the idea of search in a **problem space** consisting of various **states**, with **operators** providing transition from one state to another. Of course, these are the general principles at the basis of AI search of the type used by heuristic search planners like FF (Chapter 2.2.1).

Other concepts borrowed by artificial planners from human ones include plan adaptation (Berger and DiBattista, 1993; Bettina, 1999) and the use of heuristics in search (Klahr, Fay, and Dunbar, 1993). In what case-based planning is concerned, it has already been explained (Chapter 2.3) that this type of planning has been designed specifically to emulate analogy-based reasoning as conducted by human problem-solvers (Hammond, 1990; Keane, 1996; Spalazzi, 2001).

Still, human problem-solving approaches remain elusive. Human planners skip steps in the planning process (Koedinger and Anderson, 1990), and, at times, they are not even fully aware of their own planning processes/plans (Dillard, 2008): while it can be argued that artificially intelligent systems are never actually aware

of anything, their stages of the planning process are precisely defined and delimitated, and the generated plans or partial plans are clearly formulated and immediately available for inspection, rather than being vague mental constructs.

Of course, machine problem-solving does not have to mimic human problem-solving in order to be successful. People are certainly not infallible problem-solvers. The effectiveness and efficiency of human planning can be hindered not only by insufficient information or problem-solving experience, or a limited memory (all of which have some type of correspondent in machine problem-solvers), but also by factors not as obviously related to planning, such as shyness and nervousness (Berger and Bell, 1988), fatigue, illness, and rival claims upon the planner's attention.

Even more subtle weaknesses exist. While the creation of analogies, from the passable to the brilliant, in everything from low-brow daily conversation to science and high art is something we may see as fundamentally human (an assumption at the basis of the analogy-based case-based planning framework), it has been shown that people are not always adept at identifying analogies that reflect high-level relations and are deep, rather than superficial (Dunbar, 2001). Specifically, this weakness manifests in experimental contexts in which subjects are provided with a source and target object, and required to find analogies between the two (rather than producing the complete analogy themselves, i.e. choosing source and target objects as well as identifying similarities). The reason for this has been

hypothesized as being that typical experimental conditions are not conducive to knowledge encoding and retrieval based on deep, relational features, rather than superficial ones.

While it is true that diversity-aware automated problem solving (as explored herein) is inspired by human problem-solvers, who, at their best, are flexible, adaptable, and open to out-of-the-box solutions (Berger, Guilford, and Christensen, 1957; Keane, 1996; Mumford, Schultz, and Van Doorn, 2001), human can also fall prey to rigidity and reticence to consider alternative courses of action. Lack of experience, stress, tiredness, environment conditions, and other factors, as described above, can contribute to problem-solving difficulties including the failure to consider possible alternative solutions. Also, human problem-solvers may be faced with spaces of possible solutions that are too vast for them to explore exhaustively.

While AI planning was initially based on the idea of emulating the human cognitive activity that is its namesake, work like that presented herein has a different focus: that of supporting human problem-solving or offering an alternative to it, rather than faithfully replicating its underlying processes. Out of the four categories of definitions of artificially intelligent systems of Russell and Norvig (2009), this approach falls under that of acting rationally (rather than acting or thinking like humans, or thinking rationally). The objective is useful behavior that

can compensate for human problem-solving weakness or take over when human problem-solvers are unavailable.

In application domains characterized by large search spaces, diversity-aware problem-solving systems can help human users by intelligently guiding navigation through the solution space, highlighting viable, meaningfully-different alternative solutions. Such support could, for instance, be incorporated into mixed-initiative planning: an automated diversity-aware planning system could present its human collaborator with meaningfully different alternatives for achieving a specified set of goals/subgoals.

Here is another, more specific, example. The DivNDP algorithm (Chapter 6) produces policies (solutions to nondeterministic planning problems), which can be seen as the equivalent of finite-state machines (Houlette and Fu, 2003), in that they describe transitions, brought about by actions/events, between different states of a system. Finite-state machines are a commonly-used model for artificially-intelligent non-player characters in computer games. A system like DivNDP that automatically produces multiple, diverse policies/finite-state machines modeling game-character behavior can relieve game designers of some of the effort pertaining to creating vast game worlds populated by numerous characters.

Human expertise can still be put to use, even in completely automated diverse planning, through human-defined qualitative distance metrics. Alternatively, automatically learning distance metrics (a proposed extension to the work

presented herein, see Chapter 10.1) is a possible way of coming up with good, meaningful distance metrics without human intervention and effort.

Finally, I will make a brief note of the fact that computer games, as used as a testbed herein, have been used not only in Artificial Intelligence, but also in Cognitive Science, to study human learning and problem-solving (Ko, 2002).

3　Solution Diversity

In the context of this work, **diverse solution generation** means producing multiple solutions (plans or policies) to the same planning problem, with the specific intention of ensuring that the **diversity** of this set of solutions is high. I will introduce the formal definitions for solution-set **diversity** and for **diverse solution generation systems** after necessary preliminaries.

Definition 17. Let π and $\pi' \in \Pi$ be solutions to a planning problem P. A **solution distance metric** $D(\pi,\pi')$, $D: \Pi \times \Pi \rightarrow [0,\infty)$, or $D: \Pi \times \Pi \rightarrow [0,1]$ if the distance metric has been normalized, is a measure of the dissimilarity between π and π'.

Distance metrics can be either **quantitative** or **qualitative**, as described in Chapter 3.2. It is important to note that the problem of comparing plans and policies can be a complex task: each plan/policy may have an arbitrary number of actions/state-action pairs, each action with any number of parameters. Also, it is difficult to infer meaningful differences from the low-level structure of plans and policies.

Definition 18. Let Π be a set of solutions (plans or policies) to a planning problem. The **diversity** $Div(\Pi)$ of Π is the average pair-wise distance between solutions in Π (Equation 2).

$$Div(\Pi) = \sum_{\pi, \pi' \in \Pi} \frac{D(\pi, \pi')}{\frac{|\Pi| \times (|\Pi| - 1)}{2}} \qquad (2)$$

In Equation 2, π is a solution to a planning problem (a plan in deterministic planning, a policy in nondeterministic planning), Π is a non-empty solution set, and *D* is a solution distance metric. Similar formulas have been used by Myers and Lee (1999) in planning, and Smyth and McClave (2001) in case-based reasoning.

Definition 19. A **diverse solution generation system** DivPL is a planning system which receives as input a planning problem *P* (either deterministic or nondeterministic) and returns as output a set Π of solutions to *P*, where one of the criteria used in generating Π is that of increasing *Div*(Π) (Equation 2).

In this work, each presented diverse planning system DivPL is based on a regular, non-diverse planning system *PL*.

Definition 20. Let π be a (complete or partial) solution π to a planning problem *P* and let Π be a set of complete solutions to *P*. The **relative diversity** between π and Π is the average distance between π and each solution $\pi_i \in \Pi$. Given solution distance metric *D*, the relative diversity *RelDiv*(π, Π) of a solution π relative to Π is defined in Equation 3.

$$RelDiv(\pi, \Pi) = \frac{\sum_{\pi' \in \Pi} D(\pi, \pi')}{|\Pi|} \qquad (3)$$

While solution set diversity (Equation 2) can be used to assess the diversity of an already generated set of solutions, relative diversity is used at planning time to evaluate candidate solutions, as will be shown in the next subchapter. Relative diversity was previously used by Smyth and McClave (2001) for creating diversity in case-based reasoning for analysis tasks.

3.1   General Framework for Diverse Solution Generation

I now describe the general diverse planning framework (Algorithm 3) encompassing the main diverse plan generation algorithms that are presented herein: the DivFF diverse heuristic-search-based planner (Chapter 4.1), the DivCBP diverse case-based planner (Chapter 4.2), and the DivNDP diverse nondeterministic planner (Chapter 6).

All these systems are based on the same idea of repeatedly generating solutions using a composite candidate-solution evaluation criterion (Equation 1, where π is a solution and Π is a non-empty set of solutions) balancing relative diversity with adequacy of the candidate solution plan. Equation 1 is repeated below for convenience:

$$EvalCrit(\pi, \Pi) = \alpha SolAdequacy(\pi) + (1 - \alpha)RelDiv(\pi, \Pi) \quad (1)$$

In Equation 1, **solution adequacy** (*SolAdequacy*) is a candidate solution evaluation criterion specific to each planning technique: e.g. estimated goal distance in heuristic-search planning and case similarity in case-based planning. **Relative diversity** (*RelDiv*) is as previously defined (Definition 20, Equation 3) and, in this context, represents the estimated diversity of the set of solutions that will be obtained by completing the current partial candidate solution (if the candidate solution is not an already-complete solution that only needs to be retrieved, as in case-based planning) and adding it to the set of previously generated solutions. By modifying the parameter α in Equation 1, one can increase the emphasis on either solution adequacy or relative diversity.

In this framework, generation of a set of k diverse solutions is conducted as indicated in the pseudocode below (Algorithm 3). First, a solution is generated using the regular (non-diverse) variant of the planning system. This variant uses only a solution adequacy criterion (SolAdequacy in Equation 1) to assess candidate solutions. Then, k-1 additional plans are generated using a modified, diversity-aware version of the planning system. This variant uses EvalCrit (Equation 1) to assess candidate solutions.

Algorithm 3: General Framework for Iterative Diverse Solution Generation

Based on Solution Distance Metrics

Input: *PL* – the regular (non-diverse) planning system, *P* – the planning problem, and *k* - the number of diverse solutions to be generated.

Output: the set of diverse solutions Π.

π - a solution (plan or policy), π_C - a candidate solution

SolAdequacyPL - the solution adequacy criterion specific to planning system *PL*

RelDiv - relative solution diversity as described in Equation 3

generateSolution(*Crit*(π_C), *P*) generates a solution for problem *P* using criterion *Crit* to assess the candidate solution π_C.

DivPL(*PL*, *P*, *k*)

1. $\Pi \leftarrow \{\}$

2. $\pi \leftarrow$ generateSolution(SolAdequacyPL(π_C), *P*)

3. Add π to Π

4. Repeat

5. $\pi \leftarrow$ **generateSolution(*α*SolAdequacyPL(π_C)+(1-*α*)RelDiv(π_C, Π), *P*)**

6.   Add π to Π

7. Until $|\Pi| = k$ solutions have been generated

The **soundness** and **completeness** of DivPL (Algorithm 3) depend on the soundness and completeness of the regular planner *PL*. Note that, as the heuristic is modified to include the diversity criterion, heuristic **admissibility** is no longer guaranteed. Only the **optimality** of the first solution in the generated set may be guaranteed, and this only if *PL* itself guarantees it.

The diverse heuristic planner of Srivastava *et al.* (2007) is subsumed by the general framework presented above, but it is only used to generate quantitatively-diverse plans, not qualitatively-diverse ones (this distinction is explained in the next subchapter).

3.2   Quantitative and Qualitative Solution Distance Metrics

I know describe two categories of solution distance metrics which can be used as the basis of solution-set diversity evaluations (Equations 2 and 3).

Quantitative solution distance metrics are domain-independent and have the advantage of not requiring domain knowledge aside from the minimal domain description specific to each planning technique (e.g. a state-transition model in heuristic search planning; a case base and, possibly, a domain-specific adaptation method in case-based planning).

Quantitative distance, therefore, does not require plan/policy elements (such as actions) to be interpreted in any domain-specific way. It follows that any two distinct solution elements are considered equally distant from one another (e.g. in a perfume-making domain, the action of adding freesia essence to a fragrance is considered equally distant from the action of adding grapefruit essence and the action of adding hyacinth essence). This approach is inflexible, as well as likely to produce misleading results: two solutions identified as distant using a quantitative, action-set metric (such as Equation 4) could be essentially similar (e.g. in combat-based games, two plans may have very little overlap in terms of the actions they include, while being both implementations of a defensive strategy).

Equation 4 is an example of a quantitative plan distance metric (a normalized version of the metric used by Fox *et al.*, 2006).

$$D_{Quant}(\pi, \pi') = 1 - \frac{common(\pi,\pi')}{\max(|\pi|,|\pi'|)} \qquad (4)$$

In Equation 4, *common*(π, π') is the number of actions that plans π and π' have in common, and $|\pi|$ is the number of actions in plan π. Distance is computed by counting the number of actions which appear in strictly one of the compared plans, then normalizing the result.

Equation 4 is a variation of the **Jaccard distance** for sets (Equation 5, where *A* and *B* are sets).

$$D_{Jaccard}(A, B) = \frac{|A \cup B| \setminus |A \cap B|}{|A \cup B|} \tag{5}$$

Qualitative solution distance metrics are based on domain-specific knowledge, thus having the potential to reflect subtler semantic differences that a human expert might take into account when comparing two solutions (e.g. even if consisting of otherwise identical actions, a plan involving first-class air-travel will, from the point of view of a budget-conscious customer, be very different from its economy-class counterpart).

Qualitative distance metrics are defined based on interpretation, using domain knowledge, of the components of solutions (e.g. in a perfume-making domain: freesia and hyacinth are both floral essences, but grapefruit is a citrus essence; in a travel domain: a first-class plane ticket is expensive, while an economy one is affordable).

As multiple qualitative distance metrics can be defined for the same domain, it is possible to vary the set of features along which one would like to see diversity (e.g. in a travel domain, variation of ticket cost, but not means of transportation). This has practical advantage over quantitative diversity.

However, with the greater potential benefits of qualitatively-diverse-solution generation, comes the greater difficulty of achieving it. Unlike quantitative diversity, which is domain-independent, qualitative diversity requires domain knowledge to be encoded and utilized. Previously, this was achieved by Myers and Lee (1999). Their approach (see Chapter 9.3) assumes the availability of a *metatheory* providing additional domain information allowing plans to be compared in terms of high-level features, such as the objects which fulfill various *roles* in plans and the domain-specific characteristics of various types of actions (e.g. the speed of travel by a given means of transportation). The approach taken herein makes no such assumption, and is based on the observation that qualitatively diverse solutions can be generated without a comprehensive qualitative model of the domain: it is sufficient to equip a qualitative distance metric with minimal knowledge regarding the select features it should base its differentiation between solutions on, and use this metric as part of the solution generation criterion. Multiple qualitative distance metrics can be defined for any domain, each metric reflecting the minimal information necessary for a particular diverse solution generation task. These metrics can then be used separately or compounded as needed.

The diverse-solution generation algorithms proposed herein can be used with any distance metric, either quantitative or qualitative. As long as they are provided with the necessary metrics, these algorithms can generate both quantitatively-

diverse and qualitatively-diverse sets of solutions, without requiring a domain metatheory.

3.3   Qualitative and Quantitative Solution Diversity in an Illustrative Domain: The *Wargus* Real-Time-Strategy Game

I now introduce Wargus, a planning domain based on characteristics of the Wargus real-time strategy game (Ontañón *et al*., 2010). It is used herein to exemplify quantitative and qualitative solution diversity, and later as a testbed for the evaluation of the proposed diverse planning algorithms. It was chosen because real-time strategy games are characterized by many of the complexities of domains of practical interest: they are dynamic, nondeterministic, and adversarial. Due to these characteristics, they make solid, challenging evaluation environments for Artificial Intelligence techniques (Ontañón *et al*., 2010). Also, there are possibilities for meaningful diversity in this domain: playable characters have varied characteristics and they are capable of engaging in activities of different types: not only battle, but also harvesting and building.

Assume the following game configuration[5]: the types of available fighting units are *peasants*, *soldiers*, *archers* and *mages*. Units vary in terms of attack capabilities (e.g. soldiers are close-combat units, archers and mages long-range attack units) and robustness (e.g. peasants are the weakest units by far).

The game score is computed by adding points for enemy kills and subtracting points for loss of units. The amount added/subtracted on the destruction of a unit depends on the type of the unit in question.

The actions that can be executed by units are: *move* (the unit attempts to move to a specified location on the map), *patrol* (the unit moves back and forth between its current location and a specified location on the map), and *attack* (the unit attacks any enemies at a specified location). The action signature is *<Action_name (parameter1, parameter2)>*, where *parameter1* specifies the unit which will undertake the action and *parameter2* specifies the target location of the action (e.g. action *Move(soldier1, loc1)* instructs unit *soldier1* to move to the map location *loc1*).

There are two teams: one is controlled using the generated plans, the other is controlled by the built-in enemy AI system.

---

[5] The actual configurations used in the experimental evaluation will vary in ways described in Chapter 8.

Assume, for exemplification, a deterministic case-based planning context, in which cases are retrieved from the case base using a retrieval criterion specifying that the set of retrieved-case solution plans should be as diverse as possible based on a given plan distance metric. Assume that the case solutions in the case base are the 3 plans in Figure 1, and assume that Plan 1 has already been retrieved by random choice, and one is now trying to find a second plan, out of the two remaining ones, that is as dissimilar as possible to Plan 1, making the resulting pair of retrieved plans as diverse as possible.

Plan 1: Move (soldier1, loc1), Move (soldier2, loc2), Move (mage1, loc3), Attack (soldier3, loc4)

Plan 2: Attack (soldier2, loc4)

Plan 3: Move (soldier1, loc1), Move (soldier2, loc2), Move (mage1, loc3), Attack (archer1, loc4)

Figure 1. Sample plans for a real-time strategy game domain. The action parameters specify the unit which will be executing the action, and the map location at which the action will take place.

First, assume that the quantitative distance metric D_{Quant} (Equation 4) is used for retrieval. The plan that is chosen is Plan 2: it shares no actions with Plan 1 (the *attack* actions in the two plans use distinct soldier units), therefore the distance between them is 1 (the maximum possible distance).

However, an informed analysis, using domain-specific information, of the individual actions would tell us that an *attack* action indicates an offensive approach to the game, while a *move* action could be interpreted in various ways: moving to a location on one's own side of the map may be considered a neutral or defensive action, while attempting to move towards the enemy side is likely offensive, indicating the intention to engage in battle. Therefore, Plans 1 and 2 may not be meaningfully different at all. They both culminate in an *attack* action at the same map location, using units, which, while distinct, are of the same type (soldiers). The three other actions that differentiate Plan 1 from Plan 2 may not be of great consequence at all, if the locations the units are moving to are on the non-hostile side of the map and not very far from the units' initial locations.

Let us now use, instead, a qualitative distance metric which considers two plans maximally distant if they attack using a different *type* of unit, and identical if they use units of the same type to attack, even if the units are distinct (this metric as well as a more complex variant of it will be used in the experimental evaluation in Chapter 8).

This new criterion assesses Plan 2 as being maximally similar to Plan 1: they use units of the same type to attack, and the other actions in Plan 1 are ignored for the purposes of the comparison, as they were not specified in the distance metric definition (this is an example of a qualitative metric including only the minimal amount of domain information that is relevant to the task at hand). As a result, the

qualitative method picks Plan 3, which attacks using an archer, a unit different from a soldier: it is long-range, weaker in close combat, and its loss leads to a different score penalty than the loss of a soldier. This makes the selected plans significantly different in light of the rules of the game.

3.4   Goal-Achievement and Inflation-Based Distance

I now define two subtypes of qualitative solution distance metrics, based on whether they reflect different ways of achieving the goals of the problem: goal-achievement distance and inflation-based distance.

Goal-achievement distance reflects differences between solutions in terms of the way in which they achieve the goals, with different solutions embodying different approaches to achieving a given goal or set of goals.

Inflation-based distance reflects solution differences that are not related to the way in which the goals are achieved.

Inflation, i.e. the addition, for the sake of increasing plan/policy-set diversity, of actions/state-action pairs which do not specifically contribute to achieving the goals, is considered undesirable when it also leads to an unnecessary increase in the size of solutions (Coman and Munoz-Avila, 2011a).

While this may certainly be the case with meaningless inflation (e.g. as will be shown, inflation created using quantitative distance metrics to generate solutions), there are various planning domains in which inflation can be beneficial. For

63

example, in planning for interactive storytelling, larger solution plans/policies may include more side-stories revealing characters' personalities and motivations, while a plan/policy execution path which is the shortest path from the initial state to a goal state may prove comparatively dull.

I once again exemplify both these types of solution distance in the Wargus domain. This time, policies (solutions of nondeterministic planning problems) are used for exemplification.

Consider a real-time strategy game environment in which character units of various types (e.g. *peasants*, *soldiers*, *archers*, *mages*) can fight enemies, harvest resources, build villages, etc. Assume that the goal is for the enemy base to be destroyed.

In such an environment, the following is a set of policies which are diverse in terms of goal-achievement.

Assume a state in which all units are alive and located near the enemy's base. For this state, the specified action is *soldier attack* in **policy 1**, *mage attack* in **policy 2**, and *archer attack* in **policy 3**.

These three policies differ solely in the way in which they achieve the goal; all three attacks are legitimate ways of attempting to reach the goal, but they are significantly different, because different unit types have different strengths and weaknesses. Note that no inflation (the addition of state-action pairs which are not essential for reaching the goal) is necessary to create diversity in this example.

Inflation-based plan/policy distance allows us to introduce side-stories which simulate compelling character variation in terms of personality traits.

For example: *policy 1* includes a detour to the country-side, where the units attack a non-hostile village (such behavior reflects ruthless personalities), *policy 2* includes building an extension to a village (indicative of kindness), and *policy 3* includes collecting a treasure (indicative of greed).

The actions which create diversity in these policies do not contribute to achieving the goal, but they can make for varied game experiences and contribute to characterization.

3.5   Example Applications of Diverse Solutions

Here is a simple illustrative example of a diverse plan set in use. Suppose we have a multi-purpose social robot that vacuums floors, but is also capable of engaging in simple interaction with the people it encounters, and of improving its interaction skills through diligent practice. The robot's current mission is to vacuum all floors in a large office environment. The general mood of the people in the environment is not known at planning time, but can be guessed right before execution.

The planning system does not generate only one plan which achieves the robot's mission, but three diverse ones. The first plan ensures that the robot encounters as many people as possible: this is useful if the mood in the office

happens to be cheerful, and people are inclined to indulge the robot's small-talk, thus helping it learn. The second plan has the robot avoid encounters, thus preventing it from becoming a nuisance to employees when the tension is high inside the office. The third plan steers the robot towards areas frequented by people generally well-disposed towards it: this strikes a balance between learning and trying not to be a source of annoyance.

If any of these plans ends in non-diagnosable failure (e.g. the chatty robot has been knocked over by an irritated employee), the robot can return to its initial state and, rather than starting again with the same plan, run the plan in the diverse set which is the most dissimilar to the one which has failed, thus having a higher likelihood of succeeding (e.g. avoid encounters, rather than seek them).

Plan diversity has been pointed out to be particularly useful in situations in which user preferences are assumed to exist, but are not explicitly provided at planning time (Nguyen *et al*, 2012), because eliciting or encoding these preferences is unfeasible or prohibitively difficult, or because preferences are weakly defined and variable (as in the fluctuating office mood example).

In military (Myers, 2006) and other adversarial planning domains, diverse plans can reflect different strategies (such as offensive vs. defensive). In plan-based intrusion-detection (Boddy *et al.*, 2005), they can raise awareness of manifold threats. Generating sets of diverse redundant plans is also a possible approach to

fault tolerance (Lussier *et al.*, 2007): if one plan fails, another, dissimilar one is more likely to succeed than a very similar one.

In computer game environments, uses of diversity-aware planning have already been introduced herein, and will be further demonstrated in the experimental evaluation (Chapter 8).

Nondeterministic-planning diversity is explored, herein, in a non-probabilistic setting, as solution diversity based on distance metrics is particularly appropriate in such a context: unlike in probabilistic planning, explicit objective functions are not assumed to be given, so one cannot obtain diversity by generating the Pareto set of solutions, optimizing several objectives, as done for probabilistic planning (Bryce *et al.*, 2007). Also, it is here assumed that planning preferences may exist, but are unknown to the planner. Nguyen *et al.* (2012) point out that generating diverse solution sets is particularly valuable in such situations.

Of course, there are situations in which, rather than diverse solutions, we are searching for solutions which are as similar as possible to a previously-generated solution which, for whatever reason, cannot be used as it is (Fox *et al.*, 2006). This necessity arises, for example, in high-risk, high-cost environments (robotic surgery and, once again, the Mars Rovers come to mind), or multi-agent environments in which commitments to other agents must be honored. Systems addressing this need have already been proposed (see Chapter 9.2).

4 Generating Diverse Solutions to Deterministic Planning Problems

Having introduced the general framework for diverse solution generation (Algorithm 3), I now describe diverse-solution-generation algorithms, which adhere to this framework, for solving deterministic planning problems. These algorithms use heuristic search planning and case-based planning techniques. As they solve deterministic planning problems, they generate **sets of diverse plans** (sequences of actions).

4.1 DivFF: Diverse Heuristic Search Planning

Heuristic search planning is particularly well-suited for modification according to the general diverse planning framework, as it generates plans by iterative refinement of partial solutions based on solution adequacy criteria, which can be adjusted to include diversity requirements.

I now present DivHPL (Algorithm 4), a general algorithm for diverse heuristic planning for deterministic planning problems, which makes use of a regular, non-diverse heuristic planning system *PL* (Coman and Muñoz-Avila, 2011a). The algorithm itself is general enough to be implemented using any heuristic-search

planner (conducting either state-space or plan-space search). Herein, it is implemented using the FF planner (Chapter 2.2.1), as DivFF.

The general diverse heuristic planning algorithm for generating a set of k diverse plans consists of the following steps:

- the heuristic-search planner *PL* (with its regular heuristic h_{PL}) is used to generate the first plan in the set;

- a modified, diversity-aware version of *PL* is run k-1 times, each time generating a plan using a composite criterion h_{mixed} (Equation 6). h_{mixed} combines two criteria for candidate selection: an estimate, based on a plan-distance metric, of the relative diversity between a candidate plan and the plan set Π generated so far, and the regular heuristic of *PL*, h_{PL}.

$$h_{mixed}(\pi, \Pi) = -\alpha h_{PL}(\pi) + (1 - \alpha)h_{diversity}(\pi, \Pi) \quad (6)$$

In Equation 6, $h_{diversity}$ is defined as in Equation 7.

$$h_{diversity}(\pi, \Pi) = RelDiv(\pi_{relax}, \Pi) \quad (7)$$

In Equation 7, *RelDiv* is defined as in Equation 3 and the plan π_{relax} is a relaxed solution of planning problem *P*, that augments π. It is constructed using the relaxed plan $\pi_{relaxPL}$, produced by *PL*. With FF (Hoffmann and Nebel, 2001), for example,

69

π_{relax} can be obtained by appending π_{relaxPL} (i.e. the sequence of actions in the relaxed domain leading from the current state to the goal state) to the current partial plan π (i.e., the sequence of actions chosen so far, leading from the initial state to the current state).

Algorithm 4: DivHPL - Diverse Heuristic-Search Planning

Input: *PL* - a heuristic search planner, h_{mixed} - a heuristic function that evaluates a candidate partial plan (Equation 6), *P* - the planning problem, and *k* - the number of diverse plans to be generated.

Output: Π, a set of diverse solution plans to the planning problem.

generatePlan(*PL*, *P*) runs an unmodified version of the planner on problem *P*.

getCandidates(*P*, π) returns the set of candidate partial solutions to problem *P* in the neighborhood of partial solution π.

selectCandidate(*C*, h_{mixed}, Π) chooses a candidate partial plan that maximizes h_{mixed}, instead of one that minimizes the regular planning heuristic, h_{PL}

DivHPL (*PL*, h_{mixed}, *P*, *k*)

1. $\Pi \leftarrow \{\}$

2. $\pi \leftarrow$ generatePlan(*PL*, *P*)

3. Add π to Π

4. Repeat

5. $\pi \leftarrow$ BalancedPlan(*PL*, h_{mixed}, *P*, Π)

6.   Add π to $\prod$

7. Until $|\prod| = k$ plans have been generated

8. Return Π

BalancedPlan(PL, h_{mixed}, P, Π)

9. $\pi \leftarrow$ empty-plan

10. Repeat

11. $C \leftarrow$ getCandidates(P, π)

12. $\pi \leftarrow$ selectCandidate(C, h_{mixed}, Π)

13. Until π is a solution for P

14. Return π

It should be noted that Equation 6 is an instantiation of Equation 1, where solution adequacy is the heuristic used by PL to assess candidate solutions. The regular planning heuristic (h_{PL}) is subtracted from the diversity metric ($h_{\text{diversity}}$) because one must seek to minimize h_{PL} (an estimate of the effort required for finding a solution), and to maximize $h_{\text{diversity}}$ (an estimate of the diversity of the generated plan set). α is a parameter used for varying the complementary weights assigned to the two criteria.

DivFF is an implementation of Algorithm 4, where *PL* is the heuristic search planner FF (Hoffman and Nebel, 2001). As explained in Chapter 2.2.1, FF conducts search in the space of possible states, and candidate states are assessed using a goal distance heuristic, which estimates the distance from the current candidate state to the goal state. The FF heuristic value of a candidate state is the length of the solution plan of a relaxation of the planning problem.

DivFF bases its evaluation of candidate states on a composite evaluation criterion taking into account both the FF heuristic value associated to the state and the estimated diversity of the set of plans that will have been generated once the current plan is completed. For the purposes of the relative-diversity assessment (Equation 3), a candidate solution plan is obtained by merging the partial plan from the initial state to the current candidate state with the relaxed-problem plan generated by FF (hence, a candidate solution is an estimate of the complete final solution).

In the context of DivFF, Line 2 in Algorithm 3 corresponds to generating a plan using regular FF, while lines 4-7 correspond to generating *k*-1 plans using the composite solution-evaluation criterion described in Equation 8 below, which is a variant of Equation 6 and Equation 1 (solution adequacy is computed using the FF heuristic).

$$EvalCritDivFF(\pi_c, \Pi) = -\alpha h_{FF}(\pi_c) + (1 - \alpha)RelDiv(\pi_c, \Pi) \quad (8)$$

In Equation 8, h_{FF} is the regular FF heuristic, π_c is a candidate plan, Π is the set of previously-generated plans, and *RelDiv* is defined as in Equation 3.

4.2   DivCBP: Diverse Case-Based Planning

It has been shown how diverse solution plans can be generated using heuristic search planning techniques. Next, the general diverse solution generation framework will be applied to case-based planning.

The goal to achieve solution diversity in case-based planning is motivated by the success obtained by incorporating diversity considerations into case-based recommender systems, which solve analysis tasks (Smyth and McClave, 2001; Shimazu, 2001; McSherry, 2002; McGinty and Smyth, 2003; Bridge and Kelly, 2006).

Similarly to the approach taken with case-based reasoning for analysis tasks, diverse solution plan sets can be obtained by modifying the set of case-based planning **retrieval criteria** to include diversity considerations. However, other characteristics specific to diversity for analysis tasks are not as easily transferable to planning, which is a synthesis task. Specifically, for analysis tasks, both the similarity and diversity retrieval criteria are applied to the problem component of

the case (which, for planning problems, would typically include the initial and final states). It is shown why this approach may be problematic in case-based planning, making it necessary to retrieve based on the diversity of solutions (plans), rather than on that of the problems (Coman and Muñoz-Avila, 2010).

Therefore, the exploration of diversity in case-based planning begins with a comparative evaluation of **state (problem) diversity** (diversity based on the initial state and/or final state) and **plan (solution) diversity** (diversity based on the solution plans).

Next, I move on to a comparative evaluation of quantitative and qualitative distance in case-based planning. Both these types of distance are demonstrated using DivCBP (Coman and Muñoz-Avila, 2011b), a diverse case-based planner based on the general diverse planning framework (Algorithm 3). DivCBP retrieves cases based on a composite criterion in which solution adequacy (a component of Equation 1) is the case-problem similarity to the new problem, the typical case-based reasoning retrieval criterion.

All the presented diverse case-based planning algorithms use a transformational-analogy (Spalazzi, 2001) case-based planning approach, in which the contents of a case are a problem (consisting of an initial and/or final state) and a solution, consisting of a plan. The new problem is defined in terms of initial and/or final state.

4.2.1   State and Plan Diversity in Case-Based Planning

Two types of diversity in case-based planning are described and assessed. Each of them targets a different component of the cases considered for retrieval:

- **state diversity** characterizes cases that are dissimilar in terms of initial and, possibly, final state, but may contain the same plan or very similar plans;

- **plan diversity** characterizes cases with diverse plans, representing alternative solutions to the planning problem.

These types of diversity are demonstrated using three case-based planning case retrieval algorithms.

The first and second algorithms are tailored specifically for state and plan diversity, respectively.

The third algorithm is diversity-aware **Greedy** retrieval, a technique previously introduced for similar purposes in recommender systems (Smyth and McClave, 2001). While, originally, it was used only for problem diversity, herein, it produces both problem (state) and solution (plan) diversity.

4.2.2   Motivation Example: State and Plan Diversity in *Wargus*

To exemplify state and plan diversity in Wargus, assume that the case base contains only three cases, representing the game situations in Figure 2, and that one is only looking for a pair of maximally-diverse cases (ignoring, for now, the criterion of

similarity to a new problem). The problems are defined in terms of the initial game state.

$$c_1 \qquad\qquad\qquad c_2$$

$$c_3$$
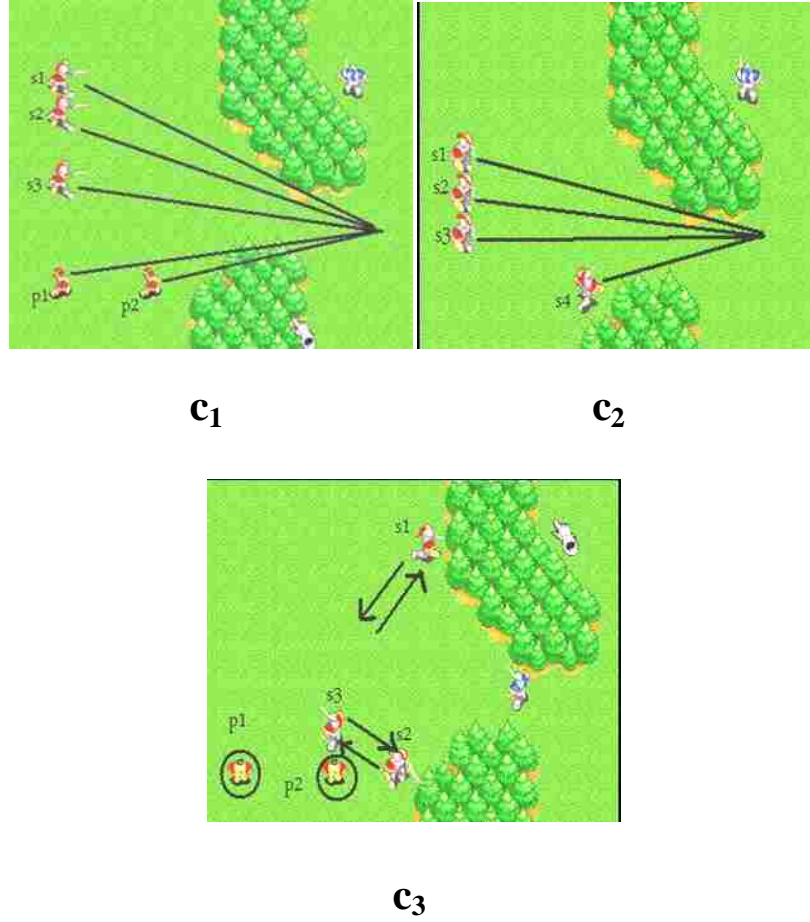
Figure 2. Case pairs (c1, c2) and (c2, c3) are state-diverse, while pairs (c1, c3) and (c2, c3) are plan-diverse. c1 and c2 are instances of the same offensive strategy (all units attack), while c3 exemplifies a distinct defensive strategy (soldier units patrol, peasant units stay put).

The three cases are c_1, c_2, and c_3, as described below.

c1: The initial state configuration includes three *soldier* units and two *peasant* units. The plan consists of all units attacking.

c2: The initial state configuration includes four *soldier* units and no *peasant* units. The plan, once again, consists of all units attacking.

c3: The initial state configuration includes three *soldier* units and two *peasant* units (it is identical to that of c_1). The plan consists of the *soldiers* patrolling given areas in expectation of an attack, while the weaker *peasant* units stay put, not "willingly" exposing themselves to injury.

c_1 and c_2 represent purely offensive strategies, whereas c_3 is defensive.

A retrieval algorithm based on state diversity (diversity of initial states only, in the experiment, for reasons explained later on) would select either **c_1 and c_2** or **c_2 and c_3** (either c_1 or c_3 would always be discarded, as they have identical initial states). Assuming, therefore, that "tie" situations such as this are handled by choosing one of the multiple cases with identical initial states randomly, in this simple example, we have a 0.5 probability of retrieving two offensive strategies, which, on being adapted to the new problem, will likely translate to solution plans reflecting identical strategies (it is reasonable to assume that an adaptation of an offensive plan will also be offensive).

If retrieving based on plan distance, however, we are guaranteed to select either c_1 and c_3 or c_2 and c_3. That is, with a probability of 1, we will retrieve two

plans that are truly different in the strategy they incorporate, and will generate distinct adaptations.

Next, I present two diverse-plan-generation algorithms specifically tailored for state diversity and plan diversity, respectively (Chapters 4.2.3 and 4.2.4). Note that these algorithms are not subsumed by the general diverse-planning framework (Algorithm 3) and they are not intended as major contributions of this work, but have only been selected so as to create state and plan diversity in the evaluation environment. It is these types of diversity that are showcased, evaluated, and compared here, not the algorithms themselves. It is important to note this, as these two algorithms are not guaranteed to produce good results on all domains and problems, although they do so in the experimental environment presented here: in these experiments, they produce maximal state and plan diversity, facilitating the comparison the two. The greedy diverse case-based planning algorithm that is subsumed by the general diverse planning framework (and is at the basis of DivCBP) will be introduced in Chapter 4.2.5.

4.2.3 State Diversity through Similarity Clusters (SDSC)

The Similarity Clusters retrieval algorithm (Algorithm 5) is designed specifically for retrieving cases which are diverse based on their problem description (states).

It works as follows.

First, cases are sorted in reverse order of their similarity to the new problem, i.e. similarity between the case problem and the new problem. Cases that are similar to one another are clustered together. To obtain a set of k cases that are similar to the new problem, as well as state-diverse from one another (so that there are no two identical state-similarity scores in the retrieved set), one case from each of the first k clusters is randomly chosen. It is assumed here that at least k clusters will be formed: this holds in the experimental environment.

Algorithm 5: SDSC - State Diversity through Similarity Clusters

Input: *newProb* – new problem, *CB* – case base, k – number of cases to be retrieved.

Output: R – the set of retrieved cases.

selectRandomCaseFromCluster(*CB*, i) selects a case from the ith cluster of case base *CB* randomly.

SDSC(*newProb*, *CB*, k)

1. $CB' \leftarrow$ cases in CB sorted in decreasing order of their similarity to *newProb*

2. $R \leftarrow \{\}$

3. For $i \leftarrow 1$ to k

4. $\quad c \leftarrow$ selectRandomCaseFromCluster(CB', i)

5. $\quad R \leftarrow R \cup \{c\}$

6. Return R

Note that it is not guaranteed that Algorithm 5 will retrieve diverse cases in all domains: cases with different similarity levels to the problem can still be similar to one another. The algorithm was chosen because it successfully produces sets of state-diverse cases in the application domain.

4.2.4   Plan Diversity through Threshold-based Selection (PDTS)

The Threshold-based Selection retrieval algorithm (Algorithm 6) is tailored specifically for retrieving cases which are diverse based on their solutions (plans).

It works as described below.

Cases in the case base *CB* are sorted in reverse order of their similarity to the new problem: *CB'* is this sorted list of cases. The case in *CB'* which is the most similar to the new problem is added to R. For each case *i*, starting with the second-highest-ranking in the hierarchy, plan diversity (*plDiv*) between it and the cases chosen so far (Equation 9) is computed. If *plDiv* is higher than a threshold Δ and similarity to the new problem is higher than a threshold Δ', then *c* is added to the retrieved set. Otherwise, the iteration stops and the retrieved cases are returned.

$$plDiv(c, R) =$$

$$\sum_{k=1,|R|} \frac{1 - plSim(c, c_k \in R)}{|R|} \qquad (9)$$

In Equation 9, *plSim* is a plan-based measure of the similarity between two cases (such as Equation 23), *c* is the current state considered for retrieval, and *R* is the list of previously-retrieved cases.

Algorithm 6: Plan Diversity through Threshold-based Selection

Input: *newProb* - planning problem, *CB* - case base, Δ and Δ' - threshold values, *k* – maximum number of retrieved cases.

Output: *R*, the set of retrieved cases.

plDiv is defined in Equation 9.

stSim is a state-based similarity metric.

PDTS(*newProb*, *CB*, Δ, Δ', *k*)

1. *CB*' $\leftarrow$ cases in *CB* in decreasing order of their similarity to *newProb*

2. *R* $\leftarrow$ { }, *i* $\leftarrow$ 2

3. Add first case in *CB*' to *R*

4. Repeat

5.   *c* $\leftarrow$ select case *i* from *CB*'

6.   If plDiv(*c*, *R*) > Δ and stSim(*newProb*, *c*) > Δ'

7.   *R* $\leftarrow$ *R* $\cup$ {*c*}

8.   *i* $\leftarrow$ *i* + 1

9. Until (plDiv(*c*, *R*) <= Δ or stSim(*newProb*, *c*) <= Δ' or *i* > k)

10. Return *R*

Note that if the similarity threshold is exceeded before k diverse cases have been identified, fewer than k cases will be returned. This does not occur in the evaluation domain.

4.2.5   Plan/State-Diversity Greedy Selection (PDGS/SDGS)

Unlike the SDSC and PDTS retrieval algorithms, the Greedy Selection algorithm (Smyth and McClave, 2001) is subsumed by the general diverse planning framework.

Algorithm 7: Plan-Diversity/State-Diversity Greedy Selection

Input: *newProb* - planning problem, *CB* - case base, *k* - number of cases in the retrieved set.

Output: *R*, the set of retrieved cases.

case is a case in the case base *CB*.

simDiv is defined in Equation 10.

PDGS and SDGS differ only based on the way in which *simDiv* is computed.

PDGS/SDGS(*newProb*, *CB*, *k*)

1. $R \leftarrow \{\}$

2.   *CB'* $\leftarrow$ all cases in *CB*

3.   For $i \leftarrow 1$ to k

4.   Sort *CB'* by simDiv(*case*, *newProb*, *R*)

5. $c \leftarrow$ top case in *CB'*

6. $R \leftarrow R \cup \{c\}$

7. $CB' \leftarrow CB' \setminus \{c\}$

8.   Return *R*

Herein, Greedy Selection is used with both problem (state) diversity and solution (plan) diversity as retrieval criteria: the two variants of the algorithm will be referred to as **State-Diversity Greedy Selection (SDGS)** and **Plan-Diversity Greedy Selection (PDGS)**, respectively.

PDGS/SDGS (Algorithm 7) retrieves a set of *k* diverse cases as follows: first, it adds to the retrieved set the case that is maximally similar to the new problem; then, for *k*-1 steps, it retrieves the case that maximizes the composite evaluation metric taking into account both the similarity to the new problem and the relative diversity with regard to the set of cases selected so far.

The Greedy Selection retrieval criterion taking into account both similarity and diversity is defined in Equation 10. Equation 10 is an instantiation of Equation 1, where *SolAdequacy* is *Sim*.

$$simDiv(c, n, R) = \alpha Sim(c, n) + (1 - \alpha)RelDiv(c, R) \qquad (10)$$

In Equation 10, *Sim* is a case similarity measure used for traditional similarity-based retrieval (state similarity), α is a parameter used for varying the complementary weights assigned to the similarity and diversity retrieval criteria, *c* is the candidate case considered for retrieval, *R* is the set of previously-retrieved cases, and *RelDiv*(*c*, *R*) is a measure of the relative diversity of a case *c* with regard to a set of cases *R*.

In Equation 11, *stSim* is a state-based measure of the similarity between two cases (such as Equation 22), *c* is the current case considered for retrieval, and *R* is the set of previously-retrieved cases.

For PDGS, *RelDiv*(*c*, *R*) = *RelDiv*(*c*.π, *R*.Π) (Equation 3), where *c*.π is the solution plan of case *c*, and *R*.Π is the set of solution plans in the set of cases *R*.

For SDGS, *RelDiv*(c, R) = *stDiv*(c, R) (Equation 11).

$$stDiv(c, R) = \sum_{k=1,|R|} \frac{1 - stSim(c, c_k \in R)}{|R|} \qquad (11)$$

4.3 DivCBP: Quantitative and Qualitative Diversity in Case-Based Planning

To demonstrate plan-diversity-aware case retrieval based on quantitative and qualitative distance metrics, I use Plan Diversity Greedy Selection (PDGS), a variant of the Greedy Selection algorithm proposed by Smyth and McClave (2001), as described in Chapter 4.2.5 above. This retrieval algorithm will be incorporated into the case-based planning algorithm **DivCBP.** Although the general DivCBP algorithm for diverse plan generation using case-based planning techniques is domain-independent, the particular implementation used in the evaluation section uses a domain-specific adaptation method tailored to the characteristics of the Wargus evaluation domain. The similarity metrics used for retrieval are also domain-specific, as are, of course, the demonstrated qualitative distance metrics.

Again, the key difference between the original Greedy Selection algorithm (used for analysis tasks) and DivCBP retrieval (used for planning, which is a synthesis task) is that plan-diversity-aware retrieval takes the solution plan into account, in addition to the problem. During retrieval, the problem is considered for similarity purposes, while the solution is considered for diversity purposes. This criterion balancing problem similarity and plan dissimilarity is captured in Equation 12.

The retrieval component of DivCBP is Algorithm 7, with *simDiv* = *EvalCritDivCBP*, as defined as in Equation 12.

DivCBP retrieval is subsumed by the general framework Algorithm 3. For DivCBP, Line 2 in Algorithm 3 corresponds to retrieving from the case base the case that is maximally similar to the new problem. Lines 4-7 correspond to the repeated retrieval, for *k-1* steps, of the case that maximizes a combined evaluation metric taking into account both the similarity to the new problem and the relative diversity with regard to the set of cases retrieved so far.

It follows that, for DivCBP, candidate solution adequacy is similarity to the new problem. Thus, the instantiation of Equation 1 can be written as follows:

$$EvalCritDivCBP(c, n, R) = \alpha Sim(c, n) + (1 - \alpha)RelDiv(c.\pi,, R.\Pi) \quad (12)$$

In Equation 12, c is the candidate case, n is the new problem, R is a set of previously retrieved cases, *Sim* is a case-similarity measure used for traditional similarity-based retrieval, $c.\pi$ is the solution plan of case c, $R.\Pi$ is the set of all solution plans in R, and *RelDiv* is defined as in Equation 3.

Retrieval of a diverse set of cases is followed by adaptation. Under the reasonable assumption that adaptation produces plans which are close to the source plan, the resulting set of plans can be reasonably expected to be diverse.

5 DivCBP vs. DivFF: A Comparative Analysis

There exists a large body of work comparing plan adaptation and first-principles planning. A number of studies have shown the benefits of adaptation-based planning approaches (such as case-based planning) over planning from scratch. In addition to experimental studies demonstrating such benefits (Veloso, 1994; Gerevini and Serina, 2000; van der Krogt and de Weerdt, 2005), general frameworks have been developed to explain them (Au, Muñoz-Avila, and Nau, 2002; Kuchibatla and Muñoz-Avila, 2006).

To this series of studies, I add a comparison of plan adaptation and first-principles planning with regard to plan diversity, i.e. effectively generating multiple dissimilar solution plans for the same planning problem (Coman and Muñoz-Avila, 2012a).

DivFF and DivCBP have been introduced in previous chapters. DivCBP conducts diverse case-based planning, using a case base storing descriptions of previously-solved planning problems and their solution plans. DivFF conducts diverse heuristic-search planning, using domain descriptions based on state-transition models.

The two algorithms, therefore, differ in terms of the knowledge they require as input, and in the way planning is conducted. DivFF requires a complete state-transition model, and the search is conducted in the set of all states that can be

constructed using this model. DivCBP uses the episodic knowledge provided by a case base, which can be incrementally updated; planning is conducted by following the case-based planning cycle.

Because DivFF uses complete domain information, it has the advantage of being able to generate any plan, which makes for higher potential plan diversity. A drawback is that the search space is exponential in the number of propositions in the domain description (based on the search-space complexity of FF, Hoffmann and Nebel, 2001). Therefore, actually finding diverse solutions may require considerable planning time. Planning effort may further increase when qualitative, rather than quantitative, distance metrics are used for diverse plan generation, as their requirements tend to be more difficult to satisfy.

Another issue potentially affecting the ability of DivFF to reliably generate diverse plans is the fact that, during planning, it assesses relative diversity based on an estimate of the final solution plan (as explained in the Chapter 4.1), as opposed to a completed plan.

The search space of DivCBP is limited to the contents of a case base. This may prove an impediment if the case base is not sufficiently diverse. More precisely, an ideal case base should include not only diverse plans, but diverse plans for diverse problems, so that whichever cases are preferred for problem-similarity purposes are also likely to include diverse plans. With such a case base, DivCBP should be able to reliably produce diverse plan sets. An important reason for this is that the

diversity assessment is based on plans in the case base, which are complete. Even with smaller case bases (which may include diverse plans only for certain problems), it may be possible to increase generated-plan diversity for DivCBP by lowering the α weight (Equation 12), so that more emphasis is given to the diversity criterion.

If the retrieved cases are diverse and the adaptation criterion ensures that the adapted plans are similar to their source plans, the set of plans generated by DivCBP will be diverse.

Finally, while DivFF will often have a larger potential search space, it will not, in most cases, explore it exhaustively (at any step, it simply searches for the first state with a better heuristic evaluation than the previously-selected one, stopping as soon as such a state is found). By contrast, DivCBP will always explore its entire search space, which, in the worst case, corresponds to the entire case base. If a case with maximally dissimilar solution exists in the case base, it is more likely to be retrieved (although this is not guaranteed: it depends on whether the case satisfies the similarity criterion as well). It also follows that the number of candidate plans considered by DivCBP will always be the same, irrespective of whether the distance metric is quantitative or qualitative.

Figure 3 provides a comparative illustration of aspects of the DivCBP and DivFF planning processes for a problem in the Wargus domain. Consider problem pa, which specifies the number of friendly armies of each type (*soldier* armies,

peasant armies, etc.). Assume plan πa has already been generated, and the system is now generating a second, dissimilar plan, using distance metric $D = D_{\mathrm{Wargus}}$ (Equation 13), for computing relative diversity.

$$D_{Wargus}(\pi, \pi') =$$

$$\begin{cases} 0, if\ attackUnitsType(\pi) = attackUnitsType(\pi') \\ d, 0 < d \leq 1, if\ attackUnitsType(\pi) \neq attackUnitsType(\pi') \end{cases} \quad (13)$$

In Equation 13, *attackUnitsType*(π) is the type of units in the attacking army of plan π, and *d* is the degree of difference between two types of units, as defined in Table 1.

The search space of DivCBP (Figure 3(b)) is a case base.

When using case base A (Figure 3), DivCBP retrieves case *c*2: even if *soldiers* are quite similar to *peasants*, this is the most dissimilar case available.

When using case base B, case *c*4 is retrieved, as its plan is maximally dissimilar to πa, based on D_{Wargus}, while its problem is very similar to *p*a.

It should be noted that, even if case *c*4 were not available, case *c*5 would be unlikely to be selected: even though it also uses *mages* to attack, its problem is not similar enough to problem *p*a.

Table 1. Domain-specific degrees of distance between types of Wargus units.

Unit type 1	Unit type 2	d
Peasant	Soldier	0.50
Peasant	Archer	0.75
Peasant	Mage	0.90
Soldier	Peasant	0.50
Soldier	Archer	0.50
Soldier	Mage	0.75
Archer	Peasant	0.75
Archer	Soldier	0.50
Archer	Mage	0.50
Mage	Peasant	0.90
Mage	Soldier	0.75
Mage	Archer	0.50

As a simplification, the search space of DivFF is represented in terms of the available actions in Figure 3(b). Strictly speaking, the search space contains all

possible states. Unlike states, which are only visited once, actions can be selected multiple times.

The diversity of plans generated by DivFF will depend on the order in which actions are considered. If an attack action $a1$ using a *peasant* army is considered before an attack action $a2$ using a *mage* army, then the partial plan containing $a1$ may be deemed sufficiently dissimilar to πa to be committed to, even if $a2$ would be the choice creating maximal diversity.

Candidate solutions for DivCBP (Figure 3*(πc)*) are completed plans. Candidate solutions for DivFF (Figure 3*(πd)*) are made up of two parts: the first has been committed to, while the second is only an estimate, computed as explained in Chapter 4.1. Even though candidate plan πd might be accurately assessed as being dissimilar to πa, the finalized plan might be different from the estimate πd, and very similar to πa.

Based on these considerations, it is to be expected that DivCBP will generate less-diverse sets of plans than DivFF when its case base is small; and to outperform DivFF in terms of generated-plan-set diversity once the case base has been augmented sufficiently.

In terms of planning time, a significant bottleneck for DivFF is the preprocessing phase, which consists of parsing and **grounding** the planning problem.

Grounding consists of instantiating operators into all possible actions, by associating them with all combinations of actual parameters out of the available objects in the problem description (e.g. operator <attack(*Soldier*, *Location*)>, given the available *Soldier* objects *sol1* and *sol2*, and the available *Location* objects *loc1* and *loc2*, will be instantiated into 4 actions: <attack(*sol1*,*loc1*)>, <attack(*sol2*,*loc1*), <attack(*sol1*,*loc2*)>, and <attack(*sol2*,*loc2*)>.

Another time-consuming stage of the DivFF planning process is calculating the goal-distance heuristic value for each candidate state: this involves constructing a planning graph and extracting a solution from it.

Also, while DivCBP assesses candidate plans already provided in a case base, DivFF needs to actually generate these candidate plans. In the case of DivCBP, I expect planning time to increase as the size of the case base increases.

A comparative experimental evaluation of DivCBP and DivFF is conducted on the Wargus real-time-strategy-game planning domain (Chapter 8.4). The diversity of generated plans is assessed both by analyzing the plans themselves and by analyzing the variation of the results obtained when running the plans in the game.

While certain behavior will be ascribable to characteristics particular to the compared systems, some conclusions can be extended to diverse plan generation systems using case-based planning and first-principles heuristic search planning techniques in general.

(pa)	**Problem:** 2 soldiers, 1 peasant, 1 mage // L1, L2: map locations; goal (as provided to Div*FF*): at least one unit has attacked
(πa)	**First generated plan:** peasant patrols L1, *soldier1 attacks L2*

(b) Search Spaces (Examples)

Div*CBP*

CASE BASE *A* (3 cases)	CASE BASE *B* (5 cases)
(c1) Prob.: 2 sol., 2 pea., 1 mage **Plan:** pea1 patrols L1, *sol1 attacks L2*	**(c1) Prob.:** 2 sol., 2 pea., 1 mage **Plan:** pea1 patrols L1, *sol1 attacks L2*
(c2) Prob.: 2 sol., 2 pea., 1 mage **Plan:** *pea1 attacks L2*	**(c2) Prob.:** 2 sol., 2 pea., 1 mage **Plan:** *pea1 attacks L2*
(c3) Prob.: 2 sol., 3 pea., 3 mages **Plan:** pea1 patrols L1, *sol1 attacks L2*	**(c3) Prob.:** 2 sol., 3 pea., 3 mages **Plan:** pea1 patrols L1, *sol1 attacks L2*
	(c4) Prob.: 2 sol., 2 pea., 1 mage **Plan:** pea2 patrols L1, *mage attacks L2*
	(c5) Prob.: 6 sol., 10 pea., 6 mages **Plan:** pea3 patrols L1, *mage1 attacks L2*

Div*FF*

sol1 attacks L1	pea. attacks L1	sol2 attacks L1	mage attacks L1
sol1 patrols L1	pea. patrols L1	sol2 patrols L1	mage patrols L1
sol1 attacks L2	pea. attacks L2	sol2 attacks L2	mage attacks L2
sol1 patrols L2	pea. patrols L2	sol2 patrols L2	mage patrols L2

Example Candidate Plans for the Purposes of Computing Relative Diversity:
(πc) Div*CBP*: peasant patrols L1, *mage attacks L2*
(πd) Div*FF*: peasant patrols L1, [*mage attacks L2*]

only an estimate, may not be part of final plan

Figure 3. DivCBP and DivFF Comparison Example (the problem and plans are simplified: (pa) problem (units represent entire armies, e.g. 1 soldier = 1 army of soldiers), (πa) the plan that has already been generated, (b) example search spaces (simplified), (πc) and (πd) example candidate solution plans.

6   DivNDP: Policy Diversity in Nondeterministic Planning

So far, it has been shown how the diverse solution generation framework can be applied to solving deterministic planning problems. In this chapter, it will be shown how the framework can be used for generating diverse sets of policies which solve nondeterministic planning problems.

Different policies for the same problem assign different actions to the same state, embodying varied strategies and approaches to a given task.

In the Robot Navigation planning domain (Cimatti *et al.*, 2003), in which a robot must distribute objects in an office setting, different policies might prioritize objects differently, or avoid/prefer different rooms. While such information is not available to the planner, in the real environment represented by the planning domain, some rooms may be best avoided because they are too crowded, while particular objects may require fast delivery.

Diversity in nondeterministic planning, as approached herein, is different from the problem of generating the Pareto set of policies, optimizing several objectives (Bryce *et al.*, 2007). Multiobjective optimization is appropriate in a probabilistic planning setting, where goals are expressed as objective functions, and the costs and rewards associated with transitions and states, as well as the probability of the actions' outcomes, are known.

Herein, I assume the availability of policy-differentiating criteria (distance metrics), but not that of solution optimization criteria (objective functions, transition cost and state reward information, or action-outcome probabilities) which would allow planning problems to be encoded and solved as optimization problems. Optimization problems are better addressed with a Pareto optimality approach (Bryce *et al*., 2007), producing multiple non-dominated solutions based on optimality, rather than diversity, criteria. Furthermore, it is often unrealistic or impractical to assume the availability of comprehensive optimization information (Ghallab, Nau and Traverso, 2004).

The availability of planning preferences is also not assumed. Nguyen *et al*. (2012) point out the importance of generating diverse solution sets in situations in which planning preferences may be assumed to exist, but are unknown to the planner, and more varied solution sets are more likely to include a satisfactory choice. This is similar to the motivation for diversity in case-based reasoning recommender systems (Smyth and McClave, 2001; Shimazu, 2001; McSherry, 2002; McGinty and Smyth, 2003; Bridge and Kelly, 2006).

Figure 4 illustrates components of policies representing different strategies in the Nondeterministic Blocksworld domain of Kuter *et al*. (2008). In this version of the well-known planning test domain, the mechanical hand might fail to pick up a block.
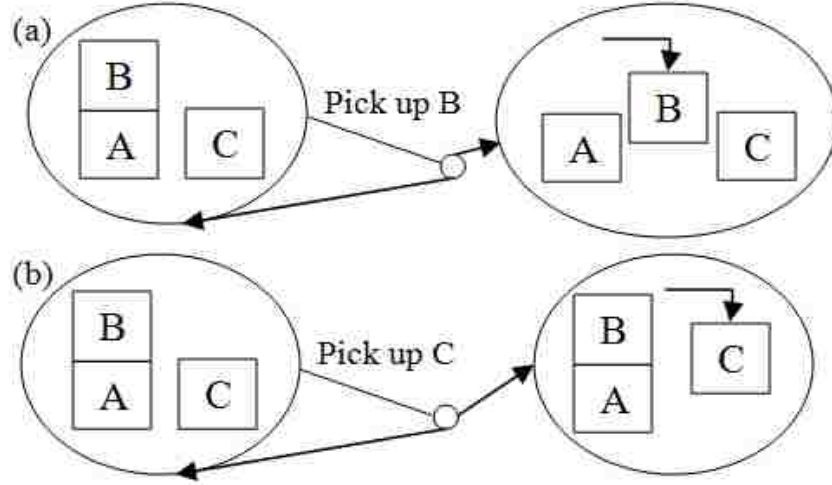
Figure 4. Sample state-action pairs for two different policies in Nondeterministic Blocksworld

Two different strategies are (a) placing all stacked blocks on the table, then stacking them in the required order, and (b) moving only the blocks which are not yet in the required position. Figure 4 presents example state-action pairs (including possible outcomes) from policies based on these two strategies. The goal is for block *C* to be on top of block *B*, and block *B* to be on top of block *A*.

The first distance metric that will be used to demonstrate policy diversity is the quantitative distance metric D_{PS} ("pair set" distance, Equation 14), a nondeterministic-planning variant of the D_{Quant} (Equation 4) quantitative distance metric used previously. In deterministic planning, quantitative distance metric values increase with the number of actions which appear in strictly one of the two compared plans. In the case of policies, the differentiation generally needs to be

based on **state-action pairs**, not just actions, because two policies which contain the exact same actions mapped to different states can lead to very different behavior at runtime.

$$D_{PS}(\pi, \pi') = \frac{|\pi \backslash \pi'| + |\pi' \backslash \pi|}{|\pi| + |\pi'|} \tag{14}$$

In Equation 14, π and π' are policies, $|\pi|$ is the number of state-action pairs in policy π, and $(\pi \backslash \pi')$ is the set of state-action pairs which are in π, but not in π'.

6.1   Generating Diverse Policies with DivNDP

DivNDP (Algorithm 8) is a diverse planning system which incorporates the NDP algorithm (Kuter *et al*., 2008) described in Chapter 2.4.1. It generates diverse sets of strong-cyclic solution policies to fully-observable nondeterministic planning problems, and can be used with any policy distance metric.

DivNDP generates *k* diverse policies as follows: it generates one policy using regular NDP and *k*-1 additional policies using a version of NDP that calls a version of *PL* with a modified heuristic function, h_{Div} (Equation 15), which balances between finding candidate partial policies that are more distant from previously-generated policies and completing the partial policies quickly.

Algorithm 8: DivNDP - Diverse Nondeterministic Planning

Input: *Dom* - a nondeterministic planning domain, S_0 - the set of initial states of the nondeterministic planning problem, *G* - the set of goal states of the nondeterministic planning problem, *PL* - the heuristic planner, and *k* - the number of diverse policies to be generated.

Output: Π, a set of diverse policies.

PLDiv(Π) is a version of *PL* which uses the heuristic function h_{Div} (Equation 15).

DivNDP(*Dom*, S_0, *G*, *PL*, *k*)

1. *Dom'* ← a deterministic relaxation of *Dom*

2. Repeat

3. $\pi \leftarrow \varnothing$; $S_0 \leftarrow S_0 \setminus G$

4.   If $S_0 = \varnothing$; then Add π to Π

5.   Else, Loop

6.   If $\exists s$ in S_0 s.t. $A_{\text{Dom}'}(s) = \varnothing$; then

7.   Return FAILURE

8. $S \leftarrow$ {all non-goal leaf states in $\Sigma\pi(S_0)$}

9.   If $S = \varnothing$; then

10. $\pi \leftarrow \pi \setminus \{(s, a) \in \pi \,|\, s \text{ is not a } \pi\text{-descendant of } S_0\}$

11.   Add π to Π

12.   Else, arbitrarily select a state $s \in S$

13.   Call *PL/PLDiv*(Π) on (*Dom'*, *s*, *G*)

99

14.   If *PL*/*PLDiv*(Π) returns a solution plan *p*

15.   π ← π ∪ {(*s*, *a*) | *a*$_i$ is an action in *p*, *a*

            is the non-deterministic action corresponding to *a*$_i$, and *s* is

            the state in which *a*$_i$ is applied in *p*}

16.   Else //*PL*/*PLDiv*(Π) returns FAILURE

17.   BACKTRACK

18. Until |Π| = *k* policies have been generated or FAILURE

19. Return Π or signal FAILURE

$$h_{\text{Div}} = -\alpha RelDiv(\pi_{\text{p}}, \Pi) + (1-\alpha)h_{\text{PL}} \qquad (15)$$

Equation 15 is a diverse nondeterministic planning heuristic, usable with any type of policy-distance metric. It is a variant of the general composite evaluation criterion (Equation 1). *RelDiv* is a measure of the relative diversity between a partial policy π_{p} (Equation 16) and the set of previously-generated policies Π, h_{PL} is the regular goal-distance heuristic of planner *PL*, while α is a parameter allowing adjustment of the weights assigned to policy diversity and goal distance, balancing diversity and performance requirements. The regular *PL* heuristic h_{PL} must be minimized, while the diversity *RelDiv* must be maximized, hence the corresponding signs (assuming *PL* seeks to minimize heuristic values).

$$\pi_p = \bigcup_i (P(p_i)) \cup P(p_p) \tag{16}$$

In Equation 16, p_i are the complete plans generated in previous runs of *PL*, p_p is the current partial plan as computed by *PL*, while $P(p)$ is the set of state-action pairs (s, a), where a is the nondeterministic action corresponding to deterministic action a_i in p, and s is a state in which a_i is applied in p.

I also introduce Equation 16', a simplified version of Equation 16, which allows diversity to be computed based only on the current partial plan p_p, instead of the entire partial policy π_p. This helps avoid redundant work when using a relative distance metric for which $h_{\text{Div}} = h_{\text{Div}}$', such as D_{PS} (Equation 14).

In Equation 16', *RelDiv'* is the relative diversity between a partial plan p_p and a set of policies Π (Equation 17).

In Equation 17, p_p is a partial plan, Π is a set of policies, $|\Pi|$ is the number of policies in Π, π is one such policy, and *RelD* is a distance metric, such *RelD*$_{\text{PS}}$ (Equation 18), indicating the degree of difference between a plan p and a policy π.

In Equation 18, *length(p)* is the number of actions in p, and $P(p)$ is a collection of state-action pairs as defined for Equation 16.

$$h_{\text{Div}}' = -\alpha RelDiv'(p_p, \Pi) + (1-\alpha)h_{\text{PL}} \tag{16'}$$

$$RelDiv'(p_p, \Pi) = \sum_{\pi \in \Pi} \frac{RelD(p_p, \pi)}{|\Pi|} \qquad (17)$$

$$RelD_{PS}(p, \pi) = \frac{|P(p) \setminus \pi|}{length(p)} \qquad (18)$$

For the purpose of the comparison between partial plan p_p and policy π, only state-action pairs (s, a) which appear in $P(p_p)$ are taken into consideration, in order to avoid shorter partial plans being evaluated as more dissimilar to π. Furthermore, pairs which do not appear in $P(p_p)$ but appear in π are less relevant for the distance computation, as state-action pairs may be added to $P(p_p)$ later on in the planning process, while the policies are already complete. In preliminary experiments, taking into account all state-action pairs in π, irrespective of whether they appear in $P(p_p)$, led to a significant decrease in the diversity of the generated policy sets.

The state-action pairs in $P(p_p)$ contain the original nondeterministic actions, not their deterministic versions in p_p: any two deterministic actions a_i and a_k originating from the same nondeterministic action a are considered identical for the purpose of the comparison between p_p and π.

6.2   Search Space Limitations and Diversity

As previously explained, heuristic-search planners increase search efficiency by reducing the search space through filters and heuristic-based candidate-solution selection. As shown in Chapter 4.1, the FF planner (Hoffmann and Nebel, 2001) evaluates candidate states heuristically using, as goal distance, the length of a solution plan to a relaxed problem obtained by ignoring negative effects of actions. When its primary search type, Enforced Hill-Climbing (EHC), which is efficient, but not complete, fails to find a solution, FF switches to complete Best-First Search, guided by the same heuristic. A filter called *Helpful Actions* is used to filter out the less promising candidate states.

However, the use of such filters and heuristic-based limitations can severely limit opportunities for creating diversity, in deterministic and nondeterministic planning alike. In Chapter 5, this has been pointed out as a weakness of DivFF, a diverse heuristic planner based on FF. In the experimental evaluation (Chapter 8.4), the diverse heuristic planner was rarely able to produce maximally diverse plan sets. This occurs because EHC never considers certain candidate states, eliminating opportunities for diversity. For the majority of the DivNDP experiments, having noted that this weakness severely impedes diversity, it has been eliminated by not using FF filters and EHC search, and relying solely on complete Best-First Search, which is also guided by a heuristic metric for ordering the candidate states, but which eventually considers the entire search space if necessary.

7 Game Character Diversity: An Application of Diverse Solution Generation in Planning

Having described algorithms for generating diverse solutions in deterministic and nondeterministic planning, I now show how these techniques can be used to enhance the diversity of non-player characters in computer games.

Character diversity can be attained by modeling game characters in terms of personalities, morals, emotional states, and social affinities. Such character modeling has been explored extensively (Cavazza, Charles, and Mead, 2002; Cavazza and Charles, 2005; Strong *et al.*, 2007; McCoy *et al.*, 2010; Cavazza *et al.*, 2009; Thue *et al.*, 2010).

An alternative method for creating diverse characters (Coman and Muñoz-Avila, 2012b) is based on behavior diversity, where character behavior is represented as plans or policies, and plan/policy diversity is created using the diverse planning algorithms described previously (DivCBP and DivNDP will be used for this purpose in the experimental evaluation in Chapter 8). In game scenarios, it will be shown how this approach creates varied character behavior reflecting different personality traits. This is a bottom-up approach to attaining character diversity: while character modeling endows characters with traits which determine their behavior (a top-down approach), plan diversity creates diversity in

terms of behavior and, as different types of behavior reflect different personality traits, this simulates personality variation.

To sum up, a character's behavior, represented as plans, reflects the character's personality traits, hence plan diversity can be the basis of character diversity.

7.1   Example Scenario

I exemplify this approach to creating character diversity in a Wargus scenario. In real-time strategy games like Wargus, characters are typically endowed with abilities such as attack strength, technical skill, endurance, etc., but they do not have individual personalities modeled in terms of temperament, morals, etc. With plan diversity, one can, to some extent, simulate character personality variety, which is a crucial component of many game genres, including role-playing games (RPGs).

Assume, hence, that the aim is to generate plans to be acted out by characters in an RPG game scenario, simulated through Wargus.

In terms of unit category, each character can be a *peasant*, *soldier*, *archer* or *mage* (unit category determines abilities, but not personality and actual behavior).

The map (Figure 5) contains buildings and units belonging to a friendly side and to an enemy side as well as neutral ones, including: a treasure, an enemy town hall, an abandoned guard tower, and a friendly peasant who is trapped in an enclosed space protected by a strong enemy paladin. With regard to his/her attitude

towards collecting the treasure, a character can be *greedy* or *not greedy*; with regard to his/her attitude toward the friend in need, the character can be *helpful* or *indifferent*. Furthermore, a helpful character can behave in a *reckless* or an *intelligent* manner in approaching their friend's rescue.
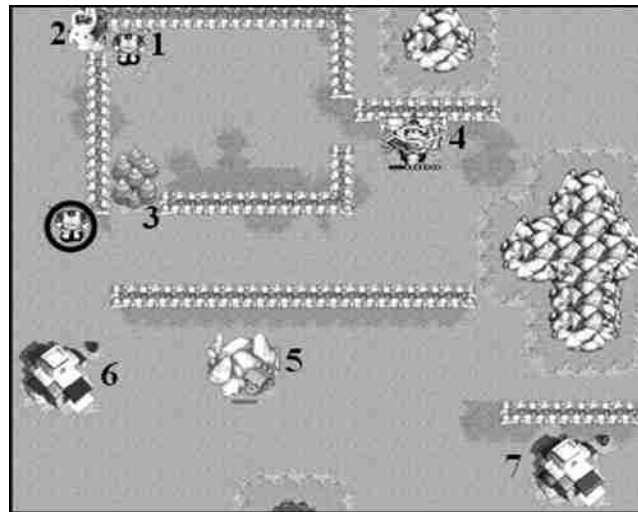
Figure 5. Example Scenario. Our character is highlighted. Other map elements of interest are: (1) friend in need, (2) old guard tower, (3) trees blocking the escape path, (4) ruthless enemy paladin, (5) treasure, (6) friendly town hall, (7) enemy town hall.

A character is *greedy* if s/he collects the treasure or loots the poorly-guarded enemy town hall (of course, the designation of this behavior as "greedy" is a choice which can change from story to story: one might instead choose to consider collecting the treasure as indicative of diligence or good citizenship); *helpful* if s/he attempts to save the captive friend. A helpful character might be considered

reckless if s/he attacks the unbeatable enemy paladin, and *intelligent* if s/he frees the friend by either cutting down trees or demolishing the old guard tower to make a path for the friend to escape by. Any other actions carried out in the plan are considered irrelevant for the purposes of this classification (e.g. it does not matter if the character takes a shortcut or the long way to the treasure). *Greed*, *bravery* and *intelligence* are, therefore, the traits along which variation will be created. A player exploring the game world populated by such character types would now encounter non-player characters (NPCs) which, while being part of the same unit category (e.g. *peasants*), exhibit varied behavior simulating personality differences.

$$D_{Char}(\pi, \pi') =$$

$$\begin{cases} 0, if\ CharType(\pi) = CharType(\pi') \\ d, 0 < d \leq 1, if\ CharType(\pi) \neq CharType(\pi') \end{cases} \quad (19)$$

To generate diverse plans for this example scenario, assume the distance metric in Equation 19 is used: π and π, are plans, and *CharType*(π) is the *type of character* (not to be confused with the unit category: unit categories are *archer*, *soldier*, etc.) reflected in plan π, while *d* is a degree of distance between possible character types. The six character types used for exemplification are: (1) *greedy* and *indifferent*, (2) *greedy* and *intelligent*, (3) *greedy* and *reckless*, (4) *not greedy*

and *indifferent*, (5) *not greedy* and *intelligent*, (6) *not greedy* and *reckless*. Equation 19 is extensible to include additional character types.

Assume that the diverse case-based planner DivCBP (Chapter 4.2) is used to produce sets of plans which are diverse based on this distance metric. The problem description consists simply of the *unit category* that the character belongs to (this information is included in the initial state), with the 4 options being: *peasant*, *soldier*, *archer*, and *mage*.

Assume that all 4 unit categories are represented in the case base. In addition to the problem description specifying a unit category, each case contains a plan that the unit is able to execute (e.g. for a *peasant* unit, the plan might specify moving to a certain location on the map and cutting down trees, then moving to another location and collecting a treasure). There are differences between units in terms of the ways in which they are able to manifest certain character traits (e.g. since no units but the *peasants* are able to harvest treasure, *soldiers*, *archers* and *mages* manifest their greed by attacking the enemy town hall).

Assume that the aim is to generate 3 diverse behavior plans for a *peasant* unit (Figure 6) and that we are using DivCBP with unit type as the similarity component of the retrieval criterion.

First, a case is selected from the case base using solely the similarity criterion, i.e. the new problem description should match the case problem description, therefore the acting unit in the case should be a *peasant*.
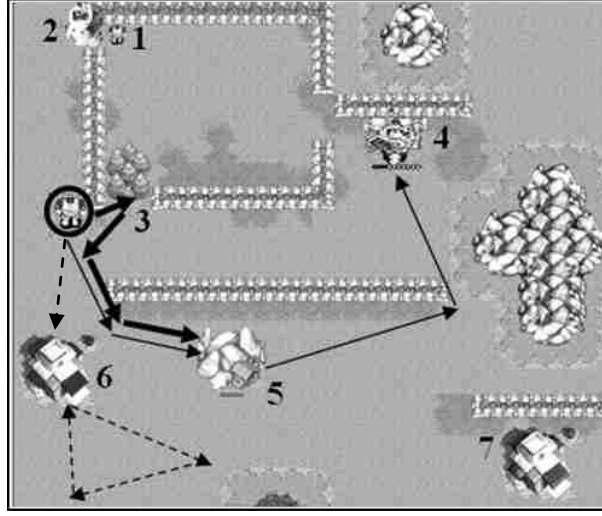
Figure 6. Potential plans that a character can execute

Out of the retrieved *peasant* cases, case c_1 is picked randomly, and ties are broken by random selection. Assume that $c_1.\pi$, the plan of the randomly picked case, specifies that the character should *collect* the nearby *treasure*, then *attack* the enemy *paladin*, in an attempt to save their friend (the plan shown by the thin arrows in Figure 6). According to the description, this plan reflects *greedy* and *reckless* behavior.

Case c_2 is now selected based on the composite criterion combining problem similarity and plan diversity with respect to $c_1.\pi$. Based on the definition of d in Equation 19, the selected plan, $c_2.\pi$, is a *not greedy* and *indifferent* character's plan: the peasant spends his/her time roaming the countryside, attacking irrelevant map locations or visiting the town hall, ignoring both the friend's predicament and the tempting treasure (see dashed arrows in Figure 6).

Finally, case c_3, another *peasant* case, is selected. Its plan should be as dissimilar as possible from both $c_1.\pi$ and $c_2.\pi$. Again, any ties are broken by random selection. The selected plan could be a *not greedy* and *intelligent* one, or a *greedy* and *intelligent* one (see thick arrows in Figure 6 for the latter).

There is now a set of three retrieved plans reflecting meaningfully different character behavior.

Of course, such a successful selection is conditioned by the availability of plans corresponding to all these character types in the case base, and by the adequacy of the plan-distance criterion when it comes to capturing meaningful behavior-related differences between plans.

In an actual game-play session, on an extended map, the three plans above might be acted out by three *peasant* characters with different personalities.

8   Experimental Evaluation

This chapter is dedicated to the experimental evaluation of the previously-described systems for diverse solution generation. DivFF is evaluated in Chapter 8.1, while DivCBP and other diversity-aware case-based planning systems are evaluated in Chapters 8.2 and 8.3. Chapter 8.4 is dedicated to the comparative evaluation of DivCBP and DivFF, and Chapter 8.5 to the use of solutions generated by DivCBP to create diverse computer game characters. DivNDP is evaluated in Chapter 8.6 (this evaluation also includes game-character diversity).

8.1   DivFF

DivFF uses the JavaFF (Coles *et al*., 2008) implementation of FF.

It is tested on 4 domains: the first 3 are synthetic, while the fourth is the Wargus real-time strategy game domain.

Two types of distance metrics are used to compute *RelDiv* (Equation 3). For the synthetic domains, a quantitative distance metric is used. For the Wargus domain, both a quantitative and a qualitative distance metric are used (the abbreviations *Quant* and *Qual* are used to distinguish between DivFF variants using these metrics).

As a baseline, I use ε-Greedy FF, a slightly modified version of JavaFF which generates sets of k plans by occasionally injecting random diversity into the

planning process: whenever choosing between candidate states, ε-Greedy FF will, with probability (1-ε), pick a candidate state randomly. In all other cases, it will behave as JavaFF would (pick the state with the best heuristic value).

8.1.1   Experimental Setup

Planning Domains. The 3 synthetic test domains used are DriverLog, Depots, and Rovers. For creating diversity in these domains, a variant of the quantitative plan distance metric D_{Quant} (Equation 4) is used.

The fourth domain is Wargus, which exhibits many of the characteristics of real domains of practical interest, and is used herein to highlight the value of qualitative plan diversity for such domains. It should be stressed that the aim is not to produce plans demonstrating expert-gamer behavior, but to create game sessions that are diverse, providing a varied sample of approaches to the game. This can, for example, be of practical value in the modeling of AI enemies, which, to make the game environment realistic and engaging, should vary in intelligence and ability.

The games take place on a small-size map (32x32 tiles). By restricting the size of the map and the types of units used, considerable intrinsic game variation is purposely not allowed.

Plans for the Wargus domain indicate what units in a team should do when competing against the built-in Wargus enemy AI. Units can *move* from a current location to an indicated one, *attack* any enemies at an indicated location on the

map, or *guard* a location (attacking any enemy that comes within a certain range). A restriction specified in the domain description is that no two units can be occupying the same map location at the same time.

The Wargus problems that the evaluation is conducted on correspond to game scenarios that are significantly different from one another. Problem descriptions indicate the available friendly unit armies and several map locations representing waypoints to which the units can move, a subset of which are "attackable" (they can be the target of an "attack" action).

$$D_{Wargus'}(\pi, \pi') =$$

$$\begin{cases} 1, attackUnitsType(\pi) \neq attackUnitsType(\pi') \\ 0, attackUnitsType(\pi) = attackUnitsType(\pi') \end{cases} \tag{20}$$

Diverse Wargus plans are generated using on a qualitative distance metric, $D_{Wargus'}$ (Equation 20), which reflects domain knowledge: a relevant characteristic setting plans apart is the type of units used for attacking, as different units have their specific strengths and weaknesses (e.g. an archer is adept at long range attacks, but weak in close combat), and their losses lead to different score penalties.

In Equation 20, *attackUnitsType*(π) is the type of units in the attacking army of plan π.

For the Wargus domain, the *Helpful Actions* filter used by JavaFF is suppressed (thus eliminating preliminary action pruning), for both DivFF and ε-Greedy FF. The *Helpful Actions* filter only considers a limited subset of the applicable actions, potentially making it impossible to obtain qualitatively diverse plans, if the actions required for doing so are not in the subset in question. In Equation 8, the assigned weight is $\alpha = 0.8$ (thus giving more weight to the FF heuristic) for the synthetic domains, in order to increase the chances of generating a solution, as it was found empirically that, with lower values of α, DivFF generates solutions for fewer problems. For the Wargus domain, the assigned weight is $\alpha = 0.55$, which is sufficient to generate solutions for all problems, and ensure that these solutions are diverse.

8.1.2   Evaluation Methods

For the synthetic domains, the diversity of the plan sets generated with DivFF and ε-Greedy FF ($\varepsilon = 0,99; 0,8; 0,7$) is evaluated using the diversity metric *Div* (Equation 2) with the quantitative distance metric used for diverse plan generation.

In the Wargus domain, the diversity of the generated plans is tested by running them in the game and observing the variation of the built-in Wargus score (which reflects damage inflicted and incurred). DivFF Qual ($D = D_{\text{Wargus}}$) is compared with ε-Greedy FF and with DivFF Quant ($D = D_{\text{Quant}}$).

In order to assess ε-Greedy FF and DivFF on equal terms, out of the problems in each domain, results are only reported for the problems on which both planner variants were able to produce complete sets of plans and did so using only Enforced Hill Climbing (not switching to complete search), as the use of a different algorithm may, by itself, influence plan diversity (e.g. if two plans in a set are generated using different algorithms, they are more likely to differ), potentially creating bias in favor of one or the other of the compared algorithms.

8.1.3   Experimental Results

For all three synthetic domains, the diversity of the plan sets generated using DivFF is, in most instances, greater than that of plan sets generated using the three ε-Greedy FF variants, while plan generation time is comparable, as can be seen in Figure 7: each point indicates the average of the diversity or planning time (as indicated by the y-axis label) over 4 planning sessions (with k=4) on one domain problem.

It should be noted that, as ε decreases, ε-Greedy FF produces increasingly greater diversity (because more random choices are being made), but the number of failed planning attempts also increases. This causes 0.7-Greedy FF to repeatedly fail on the last two problems in the DriverLog domain, never producing results (hence, the two missing data points in the DriverLog section of Figure 7). Increasing diversity by greatly increasing ε is, therefore, not a feasible approach.

115

In-game results for the Wargus domain (Figure 8) indicate that DivFF Qual generates sets of plans containing, on average, more distinct values than both DivFF Quant and 0.7-Greedy FF.

In Figure 8, each point indicates the average game score for 4 game runs of the same plan (the fluctuation of the score is shown in the error bars).

For Problem 1, DivFF Qual produces 3 out of 4 distinct scores on the first and third plan sets, and 4 distinct scores on the second one, while DivFF Quant produces 2 out of 4 distinct scores on the first two sets and 3 out 4 distinct scores on the third set. For Problem 2, DivFF Qual produces 3 out of 4 distinct scores on the first two plan sets and 2 out of 4 distinct scores on the last set, while DivFF Quant produces 2 out of 4 distinct scores on all three sets. For Problem 3, DivFF Qual produces three maximally diverse sets of scores (4 distinct scores per plan set), while DivFF Quant produces 2 out of 4 distinct scores on the first set, 4 out of 4 distinct scores on the second set and 3 out of 4 distinct scores on the third set. For Problems 4 and 5, DivFF Qual produces 3 out of 4 distinct scores on all plan sets, while DivFF Quant produces 2 out of 4 distinct scores on all plan sets.
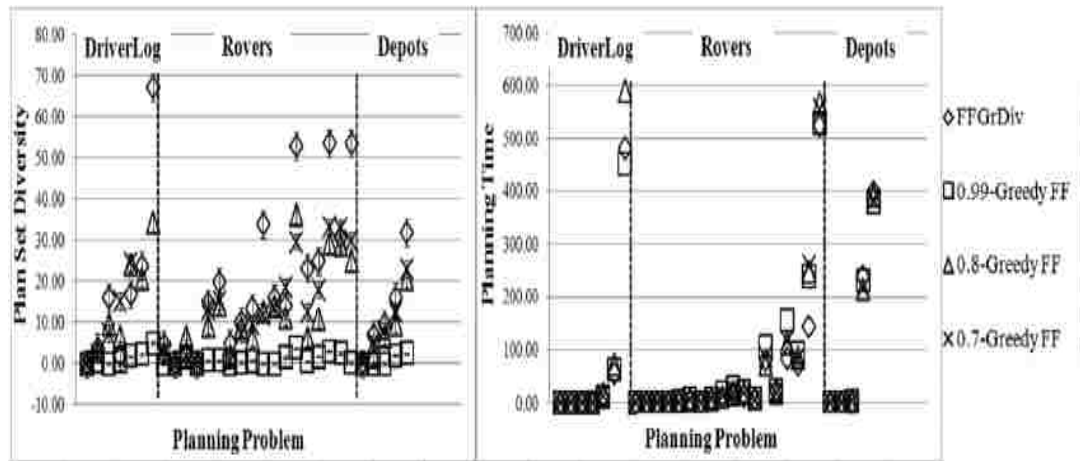
Figure 7. Synthetic domains: diversity (left) and planning time (right)
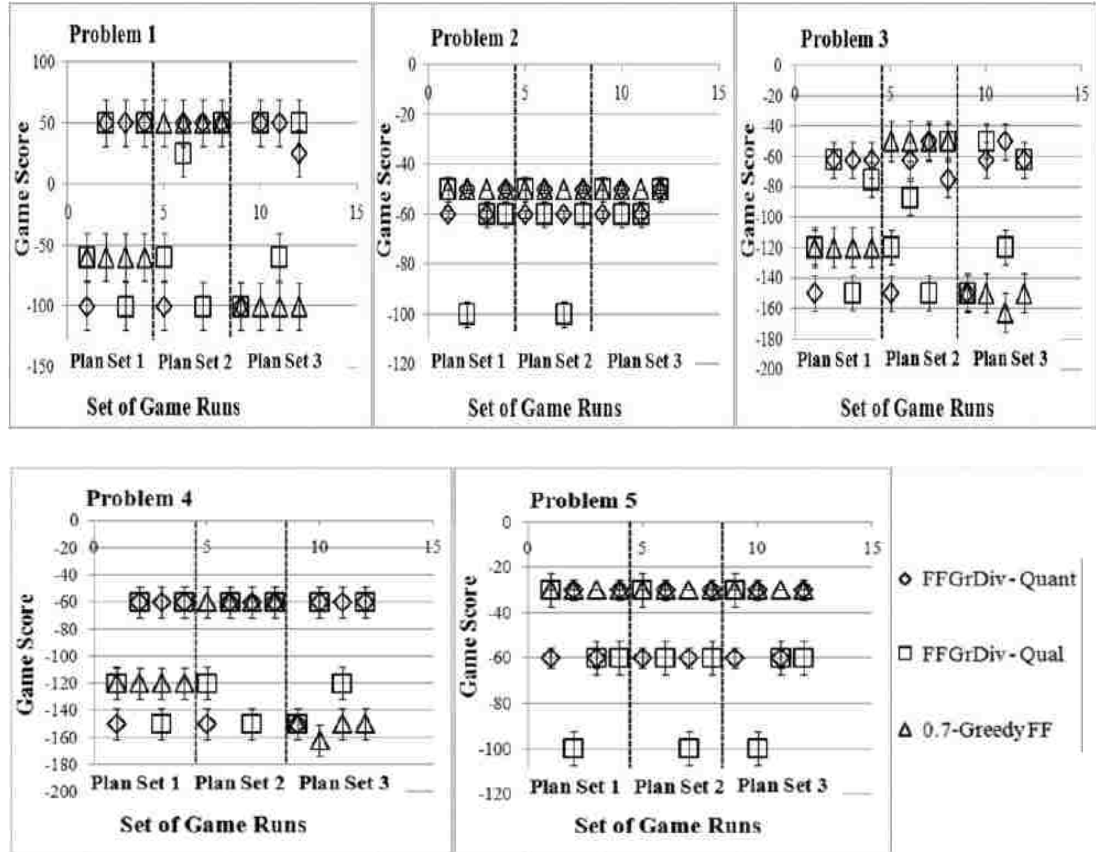
Figure 8. Wargus domain: game scores for each planning problem

117

0.7-Greedy FF only produces 2 out of 4 distinct scores on the third plan set in Problem 3 and the third plan set in Problem 4, achieving no score diversity at all in any other plan set.

In addition to showing that both variants of DivFF outperform a randomized diverse-plan generation method, the results suggest that the qualitative distance metric can, indeed, help generate plan sets of greater genuine diversity, as attested by their behavior in Wargus.

It should also pointed out that, for each problem, a subset of DivFF Qual plans perform at least as well as any plans generated with DivFF Quant or 0.7-Greedy FF (as reflected in game scores).

Furthermore, it has been observed that DivFF Quant tends to produce plans of questionable quality, by inflating them with actions not necessary for reaching the goal state, added solely for the purpose of increasing the distance to the previously-generated set of plans. In contrast, DivFF Qual generally restricts itself to adding actions which are necessary for reaching the goal state. This is reflected in the lower average length of DivFF Qual plans, and suggests that a well-chosen qualitative plan distance metric may help ensure that the generated diverse plans are also of good quality.

8.2 State Diversity vs. Plan Diversity in Case-based Planning

8.2.1 Experimental Setup

Experiments are conducted in the Wargus domain, on a 32x32 tile map. Two types of units are used: *soldiers* and *peasants*. The layout of the terrain and lack of building units ensure the brevity of game-play episodes, making them easily comparable to each other.

State similarity is based only on the initial state configuration (describing the friendly units) because plans are not annotated with their goals (e.g., capture the center of the map). This simulates a situation in which successful plans are captured by observing a player's actions, without knowing the player's intention. An initial state is defined by a pair of type *(numSold, numPeas)*, where *numSold* is the number of *soldier* units and *numPeas* the number of *peasant* units in the initial state that will be controlled through the generated plans.

Equation 21 is *stSim*, the state similarity metric used by the SDGS and PDGS algorithms.

$$stSim(c_1, c_2) = \frac{\frac{min(numSold_1, numSold_2)}{max(numSold_1, numSold_2)} + \frac{min(numPeas_1, numPeas_2)}{max(numPeas_1, numPeas_2)}}{2} \qquad (21)$$

In Equation 21, $c1$ and $c2$ are case-base cases, $numSold_i$ is the number of *soldier* units in case i, and $numPeas_i$ is the number of *peasant* units in case i.

The plans can include the actions *attackMove* (moving to a specified location on the map while attacking any enemy units encountered on the way), *patrol*, *move*, and *attack* (as described in Chapter 3.3). At this stage, no coordinates associated with these moves are taken into account when computing plan similarity. Groups of units will, as a rule, move in the same direction, both in the case-base plans and in the adapted ones.

Plan similarity *plSim* between two plans is defined as in Equation 22.

$$plSim(c_1, c_2) =$$

$$\frac{\frac{min(numAttack_1, numAttack_2)}{max(numAttack_1, numAttack_2)} + \frac{min(numMoveAttack_1, numMoveAttack_2)}{max(numMoveAttack_1, numMoveAttack_2)}}{4} +$$

$$\frac{\frac{min(numPatrol_1, numPatrol_2)}{max(numPatrol_1, numPatrol_2)} + \frac{min(numMove_1, numMove_2)}{max(numMove_1, numMove_2)}}{4} \quad (22)$$

In Equation 22, $c1$ and $c2$ are case-base cases, and $num[Action]_i$ is the number of actions of type *[Action]* in case i.

The new problem is defined by its initial state, a *(numSold, numPeas)* pair. The new problem used for experimental evaluation has the initial state (10, 9): 10 *soldiers* and 9 *peasants* (Figure 9).

Although the following information is not stored in the case base, conceptually, there are 4 plan strategies:

(1) *Offensive* (all units attack)

(2) *Defensive* (all *soldier* units patrol, all *peasant* units stay put)

(3) *Balanced Offensive* (75% of *soldiers* attack, 25% patrol)

(4) *Balanced Defensive* (50% of *soldiers* attack, 50% patrol).

The case base contains 16 cases (Table 2), with solution plans consisting of all possible state-strategy combinations between 4 start states (with varying numbers of *peasant* and *soldier* units) and a number of plans

.

Figure 9. Initial configurations of a case and a new problem

Table 2 shows only a summarization of the plans; the following is an example of an actual plan, as stored in the case base:

```
m_pb.attackMove(1,13,10);

m_pb.move(2, 7, 8);

m_pb.patrol(2, 9, 7);

m_pb.move(13, 5, 9);

m_pb.patrol(13, 6, 10);
```

The plan above instructs unit 1 to move to coordinates (13,10) on the map, while attacking any enemy unit encountered on the way; and units 2 and 13 to move to coordinates (7,8) and (5,9), respectively, and to start patrolling back and forth between their new location and coordinates (9,7) and (6,10), respectively.

Adaptation is performed by building a plan based on the same strategy as the retrieved plan, but adjusted to the number of units in the new-problem initial state. For example, if the retrieved plan is Case 2 in Table 2, a defensive plan, the adapted plan for the new problem will have all 10 *soldier* units move to a key location and patrol, while all 9 *peasan*t units remain where they are.

Each of the four retrieval algorithms (State Diversity through Similarity Clusters - SDSC, Plan Diversity through Threshold-based Selection - PDTS, State Diversity Greedy Selection - SDGS, and Plan Diversity Greedy Selection - PDGS) is run on the new problem 4 times (with tie-breaking handled by random selection), each time recording the top 4 retrieved plans ($k=4$).

For PDTS, the thresholds are $\Delta = 0.3$ and $\Delta' = 0.5$. For PDGS and SDGS, α is set at 0.5.

Table 2. Summarizations of the initial states and plans in the case-base cases

	CASES	
	Initial state	**Plan summarization**
1.	8 s, 3 p	AttackMove x 11
2.	8 s, 3 p	Move x 8, Patrol x 8
3.	8 s, 3 p	AttackMove x 6, Move x 2, Patrol x 2
4.	8 s, 3 p	AttackMove x 4, Move x 4, Patrol x 4
5.	3 s, 2 p	AttackMove x 5
6.	3 s, 2 p	Move x 3, Patrol x 3
7.	3 s, 2 p	AttackMove x 2, Move x 1, Patrol x 1
8.	3 s, 2 p	AttackMove x 1, Move x 2, Patrol x 2
9.	4 s, 0 p	AttackMove x 4
10.	4 s, 0 p	Move x 4, Patrol x 4
11.	4 s, 0 p	AttackMove x 3, Move x 1, Patrol x 1
12.	4 s, 0 p	AttackMove x 2, Move x 2, Patrol x 2
13.	5 s, 5 p	AttackMove x 10
14.	5 s, 5 p	Move x 5, Patrol x 5
15.	5 s, 5 p	AttackMove x 3, Move x 2, Patrol x 2
16.	5 s, 5 p	AttackMove x 2, Move x 3, Patrol x 3

Retrieved plans are adapted to the new problem, and the resulting sets of plans are run in the game. At the end of each such game (after all enemy units have been destroyed), the values of two evaluation metrics are recorded: **game duration** in number of game cycles (as recorded by Wargus) and **score** (consisting of the

difference between the player's score and the opponent's score, as computed by Wargus and explained in Chapter 3.3).

The *hypothesis* is that plan-diverse cases will produce greater game variation than state-diverse cases. On adapting plan-diverse retrieved cases and running them in the game, it is expected that the results (measured through game-specific metrics) will be quantifiably more varied than those obtained using plans retrieved through state-diversity methods.

8.2.2   Experimental Results

The results are shown in Figure 11 (each point in each of the graphs represents results averaged over 4 games).

Figure 10. A second map, topologically-different from that in Figure 9 (two gaps in the forest between the camps)

Incorporating the plan diversity criterion into the retrieval process leads to the selection of plans which, after being adapted and run in the game environment, generate visibly varied results (both for score and for game cycles).

The variation in results for state-diverse plans is negligible by comparison, and largely due to the random factor introduced by the tie-breaking mechanism as well as to the nondeterministic nature of the game (even when running several games with identical initial configuration and strategies, there will be some variation in game duration and final score). Diversity based on these factors is not as valuable, as its consistency cannot be guaranteed over multiple runs.

Both tested plan-diversity-aware algorithms, PDTS and PDGS, always retrieve sets of plans containing all four types of strategies, whereas, with state diversity, the retrieval of a highly diverse set is unlikely. In SDSC test runs, at least two results from the state-diverse retrieval set are always instances of the same strategy.

Each of the methods based on plan diversity produces in-game results within a larger range than its state-diversity counterpart: the highest and the lowest values, in terms of score as well as game duration, were obtained by running plans retrieved with plan-diversity-aware methods.

For plans based on those retrieved with the PDTS retrieval algorithm, the standard deviation is 211.5 for number of game cycles and 84.3 for score (compare with 29.9 for number of game cycles and 7.2 for score in the case of SDSC plans).

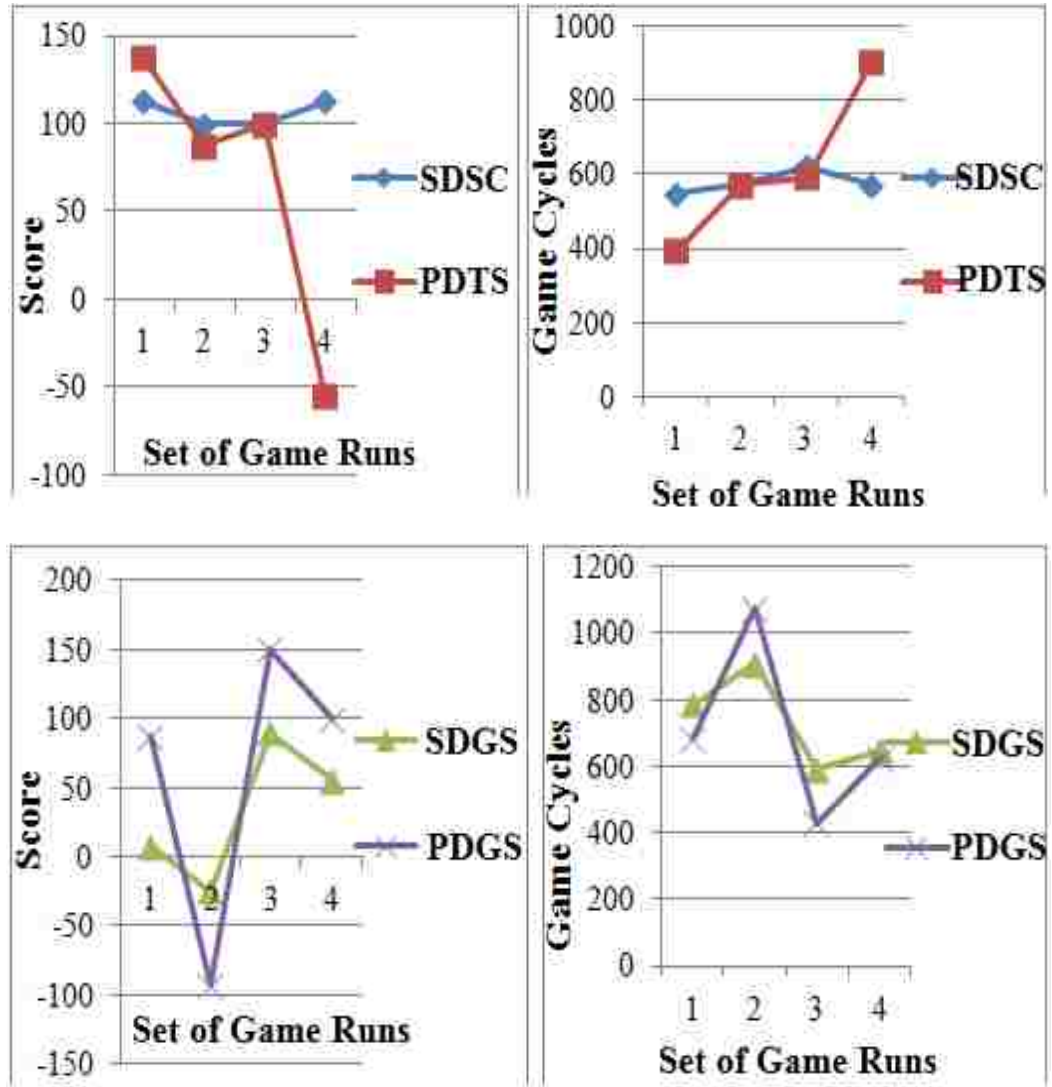**Figure 11.** (a) State-Diversity by Similarity Clusters vs. Plan-Diversity by Threshold-based Selection (b) State-Diversity Greedy Selection vs. Plan-Diversity Greedy Selection

In the case of PDGS plans, the standard deviation is approximately twice as large as for SDGS plans (271.4 for number of game cycles and 105.9 for score, compared to 142.9 for number of game cycles and 50.8 for score).

Similar results were obtained on an alternative map (Figure 10) that is topologically different from the first one, in that there are two gaps in the forest initially separating the two opponent camps, offering more passage possibilities for the enemy army.

8.3   DivCBP

8.3.1   Experimental Setup

Two-player Wargus games are run on two 32x32 tile maps (Figure 12). Compared to Chapter 8.2, the game configurations are more sophisticated (more unit types and more complex plans), as is the case-based planning system.

The types of units and available actions are as described in Chapter 3.3.

Each plan represents an individual battle (in which one of the armies challenges the enemy), rather than a complete, prolonged game. This restriction was necessary so as not to allow excessive implicit game-play diversity, which might render meaningless the difference in variance between results produced using different metrics.

Again, the two maps on which the generated plans are tested are topologically different: the first has one gap in the forest separating the two armies, while the second has two gaps, located at different coordinates than the gap in the first map. This difference is meaningful: on the second map, units will sometimes make different choices as to which gap to use to pass to the other side: sometimes, all

127

units will use the same gap, at other times, they will split up, sometimes they will even "hesitate", marching towards one gap, then returning to the other one. This ensures considerably different game behavior between the two maps.

Case-based Planning System. In the case-based planning system, the following convention is used: the **cases** are interpreted as battle-plan blueprints, so that every unit in a case is an abstracted representation of an entire army of units of that type (e.g. a soldier stands for an army of soldiers). New problems consist of actual game configurations, specifying number of armies of each type, as well as number of units in each army.

Figure 12. The two topologically-different game maps, with archer armies highlighted. Note how the archer army in the second map has split up into two divisions, each using a different gap to pass.

RETRIEVED CASE	ADAPTED PLAN
	move (archer1, 05_05)
	move (archer2, 05_05)
	move (archer3, 05_05)
Initial State	*move (archer4, 05_05)*
1 soldier army	move (peasant1, 03_02)
1 peasant army	move (peasant2, 03_02)
1 mage army	move (peasant3, 03_02)
1 archer army	move (peasant4, 03_02)
	move (mage1, 04_07)
Plan	*move (mage2, 04_07)*
move (archer1, 05_05)	*move (mage3, 04_07)*
move (peasant1, 03_02)	*move (mage4, 04_07)*
move (mage1, 04_07)	move (archer1, 24_07)
move (archer1, 24_07)	move (archer2, 24_07)
patrol (soldier1, 01_04)	move (archer3, 24_07)
move (soldier1, 05_04)	move (archer4, 24_07)
attack (archer1, 24_07)	*patrol (soldier1, 01_04)*
	patrol (soldier2, 01_04)
NEW PROBLEM	*patrol (soldier3, 01_04)*
	patrol (soldier4, 01_04)
Initial State	move (soldier1, 05_04)
1 soldier army	move (soldier2, 05_04)
(4 units)	move (soldier3, 05_04)
1 peasant army	move (soldier4, 05_04)
(4 units)	*attack (archer1, 24_07)*
2 mage armies	*attack (archer2, 24_07)*
(4 units each)	*attack (archer3, 24_07)*
2 archer armies	*attack (archer4, 24_07)*
(4 units each)	

Figure 13. Sample case, new problem, and the corresponding adapted plan (units in the adapted plan are not annotated with the army they belong to because, in this example, there is only one army of each type). The second action parameter indicates the coordinates of the location at which the action should take place.

The **case base** consists of 100 distinct cases, each composed of an initial state (the problem) and a plan (the solution).

The **initial state** is represented in terms of numbers of armies of each type. Each of these armies is represented by one unit in the plan.

The **case solution plans** were generated using the FF generative planner (Hoffmann and Nebel, 2001), modified so as to generate multiple plans for the same problem. All case-base plans contain an attack action carried out by one unit (which represents the entire attacking army in the adapted plan). No goal state is specified: the general goal is to obtain the highest possible score, and there is never one single final state through which this is achieved.

The **new problems** consist of initial game states, indicating the number of armies of each type (*soldier*, *archer*, *mage*, *peasant*) as well as the number of units in each of the armies. All units in an army are of the same type. There are 5 new problems, with varying numbers of armies of each type, as well as number of units per army.

The **adaptation algorithm** is consistent with the idea of a retrieved plan serving as blueprint. As each unit in the retrieved plan represents an army, each army A in the new problem will be matched to a unit U (of the same type as the units in A) in the retrieved plan. All units in A will then perform all actions performed by U in the retrieved plan. The matching will occur in order of the numbering of units in the retrieved plan, with one exception: if unit U is the

attacking unit in the retrieved plan, *U* will be the first to be assigned to an army of its type in the new problem, assuming such an army exists. This will always be the case with the problems used for testing: they all contain at least one army of each type, in order to be able to take at least partial advantage of any retrieved plan. An adaptation example is provided in Figure 13.

For **case retrieval**, the PDGS retrieval algorithm (Algorithm 7) is used, where $k = 4$, $\alpha = 0.5$, and *Sim* is a similarity metric Sim_{InitSt} (Equation 23) based on the initial states of the compared cases.

$$Sim_{InitSt}(c_1.\text{IS}, c_2.\text{IS}) =$$

$$\frac{\sum_{i=1}^{n} \frac{\min\left(numArmiesType_i(c_1.\text{IS}), numArmiesType_i(c_2.\text{IS})\right)}{\max\left(numArmiesType_i(c_1.\text{IS}), numArmiesType_i(c_2.\text{IS})\right)}}{n} \tag{23}$$

In Equation 23, *n* is the number of types of units (in the experimental setup, *n* = 4) and *numArmiesType$_i$(c.IS)* is the number of armies of units of type *i* in the initial state of case *c*.

As the distance metric *D*, the quantitative metric D_{Quant} (Equation 4) and the game-specific qualitative metric D_{Wargus} (Equation 13) are used.

8.3.2   Evaluation Methods

To evaluate the diversity of game-play sessions based on the sets of generated plans, the variation of two game-specific **evaluation metrics** is observed.

The primary metric is Wargus **score** (computed as described in Chapter 3.3), while the secondary metric is **time** (the duration, in game cycles, of game-play sessions).

The *hypothesis* is that plans obtained using retrieval based on the qualitative plan-diversity metric D_{Wargus} will produce greater game-play variation (reflected in the evaluation metrics) than plans obtained using the action-set quantitative distance metric D_{Quant}. It is to be expected that, when run in the game, adaptations of plans retrieved using the qualitative distance metric will produce significantly more variation (as measured using standard deviation and assessed using the F-test) of Wargus scores than adaptations of quantitatively-diverse sets of plans. Similar results are expected with regard to time, but with less confidence, as it has been observed that game duration tends to vary more between runs of the same plan, on the same map.

Note how, out of the countless possible domain-specific, qualitative distance metrics, one was chosen in accordance with the purpose of obtaining easily quantifiable diversity. Had the objective been different, one might have opted for a distance metric producing some form of diverse game behavior which is not so clearly reflected in score variation. For example, had time been chosen as the

primary metric, one might have retrieved plans which use diverse route waypoints, encouraging the variation of game duration more clearly than the variation of score.

The question might be raised whether plan sets producing highly diverse scores, from high to low (rather than all of the plans playing the game expertly) are ever of practical value. A simple example of a situation when such plans are valuable is the modeling of AI enemies, which, to make the game environment realistic (as well as not discouragingly difficult) should vary in intelligence and ability. Also, in partially unknown environments (e.g. the enemy force may vary over consecutive plans), we may benefit from experimenting with multiple diverse plans, even if some of them behaved poorly in a slightly different game configuration.

8.3.3   Experimental Results

In Figure 14, each point in each chart represents the standard deviation of score or time (as indicated) for one plan set of 4 plans, where each plan is run in the game 5 times. The two data sets in each chart correspond to results obtained using the quantitative distance metric D_{Quant} and the qualitative metric D_{Wargus}, respectively. There are 5 plan sets for each of the 5 new problems, on each of the 2 maps (50 plan sets in all).

As can be seen in the charts, for **score**, the standard deviation of D_{Wargus} results per plan set is consistently higher than that of D_{Quant} results. Being highly diverse,

the D_Wargus score sets always include the highest recorded score per problem/map combination (while D_Quant sets do not). The F-test score results indicate that the difference between the variances of the D_Wargus and D_Quant score data sets is statistically significant, at the 95% confidence level, for all problems, on both maps, with the D_Wargus data set displaying the greater variance.

For the secondary metric of **time**, the standard deviation of D_Wargus results is greater than that of D_Quant results on all but 2 of the 25 plan sets on the first map, and all but 3 out of the 25 plan sets on the second map. The F-test indicates that the D_Wargus data sets display greater variance, and the variance difference is statistically significant, at the 95% confidence level, on 4 of the 5 problems on each map.

On the second map, the difference is statistically significant (with greater variance for the D_Wargus data set), at the 90% confidence level, on the remaining problem. For the remaining problem on the first map, the variance is slightly greater for the D_Quant data set, but the difference is not statistically significant.

To sum up, D_Wargus results are significantly more diverse than D_Quant results on all problems for the score metric, and on the majority of problems for the time metric. This is consistent with the expectations.
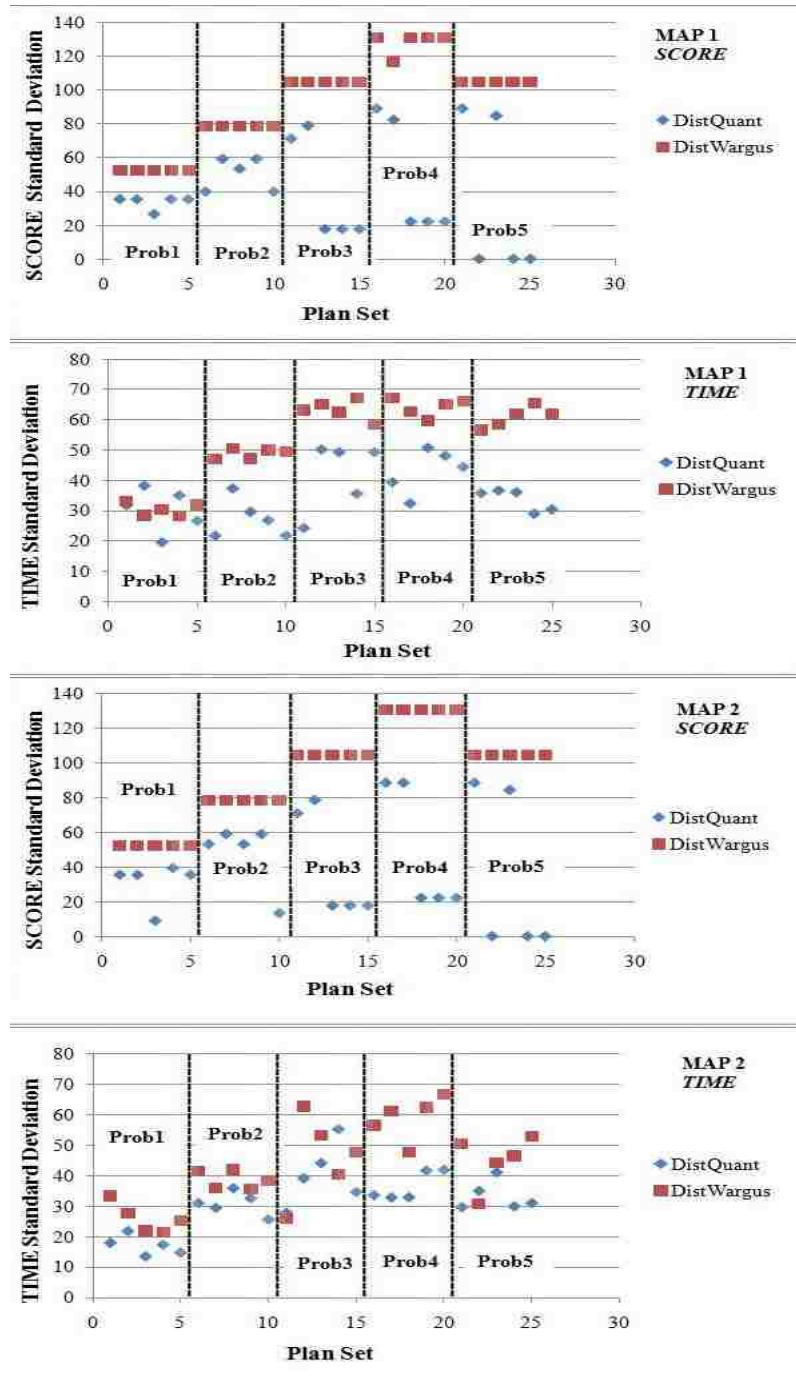
Figure 14. Standard deviation of game scores and time (game duration)

135

In terms of plan quality, it has been noticed that plans retrieved using D_{Wargus} (and, consequently, the adapted plans based on them) tend to be shorter, on average, than plans retrieved using D_{Quant} (the reason for this should be obvious from the way in which the two metrics are computed, with D_{Quant} easily increasable by lengthening any of the compared plans, as long as the added actions are not encountered in the other plan). Plan length relates to the time it takes to execute the strategy outlined in the plan. It follows that shorter plans may, in this context, be preferable to longer ones. This suggests that well-chosen qualitative distance metrics can also help ensure that retrieved plans are of good quality.

8.4 DivFF vs. DivCBP

8.4.1 Experimental Setup

The Wargus domain is once again used for evaluation. DivFF and DivCBP are used to generate battle plans for various game scenarios, varying in terms of number of armies of each type making up the team controlled through the generated plans. Both systems are used with the qualitative distance metric D_{Wargus} (Equation 13), which was shown to produce significantly higher game diversity than the baseline quantitative distance metric (Equation 4). In each session, each planner was used to generate 4 diverse plans.

Test Problems. The 5 problems in Chapter 8.3, describing varied game configurations, are used for evaluation.

For DivCBP, the game configurations specify the number of friendly armies of each type as well as the number of units in each army.

For DivFF, the configurations include the following additional information needed for the domain description: (1) the coordinates of the units in the initial state, (2) the waypoints via which fighting units can move, and (3) a goal specifying that at least one army should have attacked. While, conceptually, the goal of every game is to win, this cannot be represented through a unique state configuration, especially since enemy information is not included in state descriptions. This additional information provided to DivFF does not influence its ability to enforce plan diversity, as it is not used by the D_{Wargus} plan distance metric.

Game Configuration. The game configuration remains largely the same as in Chapter 8.3. The only change is the replacement of *archer* units with *gryphon rider* units, which make game-play sessions more varied and compelling. Unlike land units, gryphon riders move by flying, which makes them impervious to many types of attack, and often lethal if their attacks succeed. However, they can be escaped rather easily by fleeing, as they move slowly and need time to power up between attacks. This creates more game variation, irrespective of the algorithm used to generate the game plans.

Games are run on the two topologically different maps described in Chapter 8.3: both maps consist of two areas divided by an obstacle. In the first map, there is a single pass through the obstacle, whereas, in the second map, there are two passes, enabling strategic decisions that would not be applicable on the first map.

DivCBP Case Base. The case base introduced in Chapter 8.3, which contains 100 cases with plans generated using the FF planner, is used.

However, in order to put to test the expectation that the performance of DivCBP will be lower with small case bases, and will improve as the size of the case base increases, the size of the case base varies from 25 to 100 cases, thus simulating the incremental construction of a case base as more and more cases become available.

The variant of DivCBP using a case base with n cases will be referred to as DivCBPn. The cases added to the case base at each incremental increase are chosen randomly. That is, the case base is not specifically tailored towards diversity. However, as the number of cases increases (with new cases being chosen randomly), plan diversity in the case base is also likely to increase.

8.4.2   Evaluation Methods

Three categories of evaluation metrics are used to assess the performance of DivCBP and DivFF.

First, it is assessed how successful the two systems are at producing sets of plans that are diverse based on the specified distance metric. This is the standard type of **plan-set diversity** assessment conducted previously in generative planning (Myers and Lee, 1999; Srivastava *et al.*, 2007).

Second, **planning time** is reported on.

Third, to show how plan-set diversity reflects on actual **game-play diversity**, the end-game **score** and **game duration variation** obtained when running the generated plans in the Wargus domain are reported.

Next, each of these types of evaluation metrics will be discussed in detail.

Plan-Set Diversity. It is first assessed how successful the compared planning systems are at generating sets of plans that are diverse, by using the evaluation metric *Div* (Equation 2, repeated below for convenience) with distance metric D = D_{Wargus}. With this distance metric, the maximum diversity value for a set of plans is 0.65, and corresponds to a set of plans in which each plan uses an army of a different type (out of the 4 available ones: *peasants*, *soldiers*, *gryphon riders*, and *mages*) to attack.

$$Div(\Pi) = \sum_{\pi,\pi' \in \Pi} \frac{D(\pi,\pi')}{\frac{|\Pi| \times (|\Pi|-1)}{2}} \qquad (2)$$

In Equation 2, π is a plan, Π is a non-empty plan set, and D: $\Pi \times \Pi \rightarrow [0,1]$ is a plan distance metric.

The *hypothesis* with regard to this evaluation criterion is that, when equipped with a sufficiently large case base, DivCBP will be more successful than DivFF at consistently imposing the given distance metric on the set of generated plans. It is to be expected that the sets of plans produced using DivCBP with case bases of a certain size will have consistently greater *Div* scores than the sets of plans produced using DivFF.

Planning Time. Plan generation time is measured in seconds. For DivFF, preprocessing time (preprocessing is conducted only once, before all planning sessions) and planning time for each of the four generated plans are measured. For DivCBP, retrieval and adaptation time for each of the four plans are measured.

Game-play Diversity. To examine the connection between plan set diversity, assessed by analyzing the generated plans themselves, and in-game behavior, the plans are run in the game and the variation of game-play sessions is evaluated using the standard deviation of Wargus **score** and game-play **time** (the duration, measured in game cycles, of game-play sessions).

The *hypothesis* is that plan sets obtained using whichever of the two systems produces more highly diverse plan sets consistently will produce greater game-play

variation in most instances. It is to be expected that, when run in the game, plans generated using the system in question will produce more variation (as measured using standard deviation and assessed using the F-test) of Wargus score and time than plans obtained using the other system. However, game variation is also likely to be affected by the inherent nondeterminism of the game, which can cause even repeated runs of the same plan to produce different results, particularly in terms of game duration.

8.4.3   Experimental Results

Plan-Set Diversity. Plan set diversity values are shown in Figure 15. It can be observed how DivCBP (with $\alpha = 0.5$, as in Chapter 8.3) already attains maximum plan set diversity, for all problems, with a case base containing only 30 (out of 100) cases. Furthermore, once this happens, DivCBP *always* generates maximally diverse plan sets (Div = 0.65), thus confirming the expressed expectations.

On the other hand, it is also confirmed that DivFF has the advantage of being able to produce maximally diverse plan sets at any time, while DivCBP is only able to do so after the case base has become sufficiently large. However, the success of DivFF at generating maximally diverse plan sets depends largely on chance (considering the right candidate states first; while DivFF can theoretically generate all possible plans, this, in practice, is never the case). Out of the 25 plan sets it

produces for all problems, only 2 are maximally diverse; the average values per

problem attest to less successful diverse-plan-generation sessions.
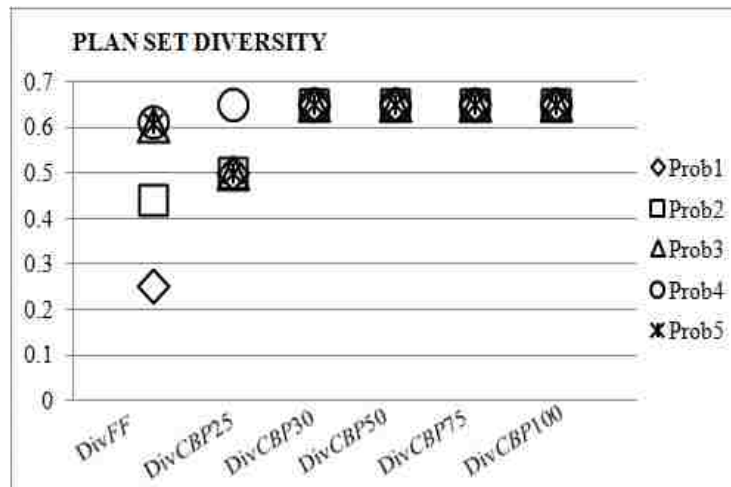

Figure 15. Plan Set Diversity (Equation 2, with $D = D_{\text{Wargus}}$) for DivFF and DivCBP with case bases of different sizes
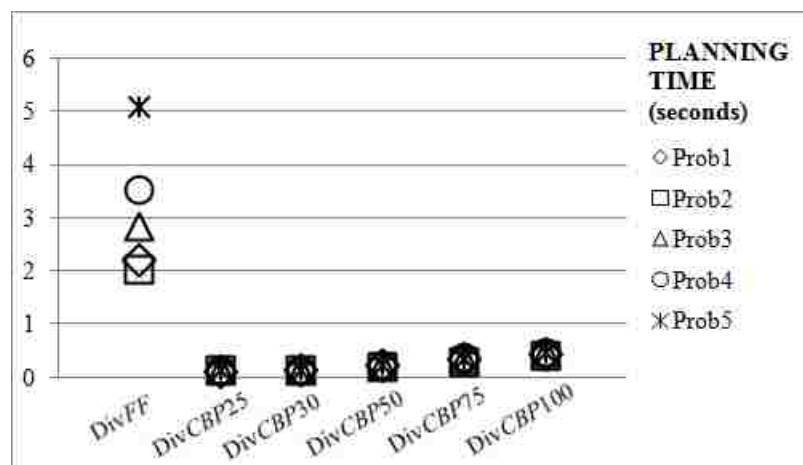

Figure 16. Planning Time for DivFF and DivCBP with case bases of different sizes

In addition, it has been ascertained that DivCBP can be encouraged to obtain greater plan-set diversity by lowering the value of the α parameter in Equation 12 (thus promoting plan diversity over state similarity). When the value of α is lowered from 0.5 to 0.35 for DivCBP*25*, the system consistently produces maximally diverse plan sets for all problems (which DivFF never achieves).

Planning Time. Figure 16 shows planning time for DivFF and DivCBP. All tested versions of DivCBP are faster than DivFF.

DivFF planning time increases steeply with the size of the initial state (i.e. the number of objects in the initial state, which in this case, consist mostly of unit armies and waypoints that the units can use for movement), due to the grounding bottleneck and to the increase in the number of states in the search space (which is a factor of the number of available objects).

DivCBP planning time is, on the set of test problems, unaffected by the size of initial states. It increases with the size of the case base, but at a slow rate.

Game Diversity. Score and time (game duration) standard deviation results are shown in Figure 17. Therein, each point represents the standard deviation of score or time (as indicated) for one plan set of 4 plans, where each plan is run in the game 5 times. There are 5 plan sets for each of the 5 problems, on each of the 2 maps (50 plan sets in all).
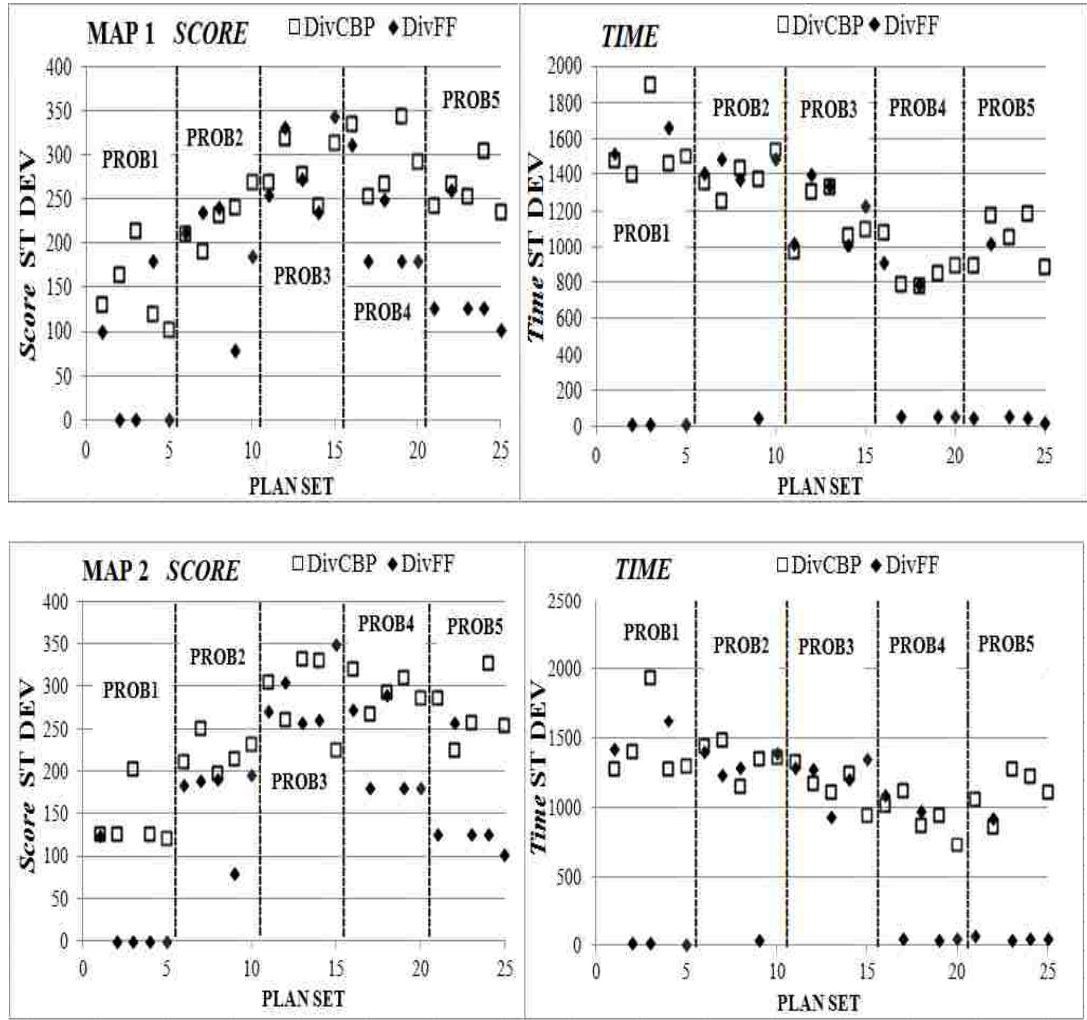
Figure 17. Standard deviation of in-game results (score and time)

The two data sets in each chart correspond to results obtained using DivFF and DivCBP*100* for plan generation. DivCBP*100* is representative of all Div*CBP* variants which always produce maximally diverse plan sets on all test problems (starting from DivCBP*30*). As these variants of Div*CBP* produce very similar

diverse plan sets, they were not tested comparatively in the game, as any difference in variation between them might more reasonably be attributed to the nondeterminism of the game, than to the particular merits of any of the variants. Beyond obtaining the maximum diversity based on the specified distance metric (which all these variants do), there is nothing more the compared planning systems can achieve in terms of diversity.

For both **score** and **time**, the standard deviation per plan set of the DivCBP results is, on average, higher than that of the DivFF results on 3 out of 5 problems (Problems 1, 4 and 5) on each map. On the remaining two problems, standard deviation is roughly the same, with no algorithm being clearly much better than the other in terms of diversity, though DivFF plans will occasionally produce results far below the problem average (which is never the case with DivCBP plans).

F-test results show that the difference between the variances of the DivFF and DivCBP score and time data sets is statistically significant, at the 95% confidence level, for the 3 problems on which the variation of DivCBP results is clearly higher, on both maps. For the remaining two problems, the difference in variation is not statistically significant on any of the maps.

8.5 Plan-Based Character Diversity

8.5.1 Experimental Setup

Game-character diversity based on diverse plans retrieved using DivCBP is showcased in Wargus, thus demonstrating character-trait diversity in an environment which does not normally ensure it.

Character diversity is based on the diversity of plans acted out by characters, where plans are represented as sequences of actions, such as *<move to location 1><attack enemy soldier><collect treasure>*. Such plans, made up of unitary actions that a character can execute, can be created easily based on player game-play traces, without the knowledge engineering required by HTN planning, which has also been used for the purposed of diversifying character behavior (Paul *et al*., 2010). In terms of domain-specific knowledge requirements, the availability of a plan repository (e.g. collected from logs of players' actions) and of user-defined plan comparison criteria (distance metrics) are assumed.

The described methods can also be used for characters operated by finite-state machines, as finite state machines also specify low-level actions which the character should execute.

To retrieve diverse plans, DivCBP is used with the distance metric D_{Char} described in Chapter 7 (Equation 19) and $\alpha = 0.5$. Retrieved plans are run in the game as they are, as adaptation is not necessary.

The case base contains 100 cases for the 4 unit categories (*peasant*, *soldier*, *archer*, and *mage* cases). While certain case-base plans are significantly different in terms of character behavior, a lot of them vary in ways which do not make for different character types (e.g. taking different paths to the same target map locations).

The game scenario is as described in Chapter 7.1.

The following possible **game outcomes**, indicating ways in which running a character plan affects the game environment, are recorded: *the gold has been collected*, *the gold has not been collected*; *the character's friend is free*, *the character's friend is captive and the character is alive*, *the character is dead*.

By recording and comparing these outcomes, one can determine whether the diverse plans that are generated actually lead to noticeably different game-play.

As experimental baseline, an alternative way of attempting to create diverse plans without any knowledge engineering at all is used: selecting plans from the case-base randomly. However, as will be shown, this does not guarantee significant diversity as reflected in game outcomes.

For each unit category, 5 sets of plans (of 6 plans each) are retrieved and run in the game. There are 30 game-play sessions per unit, in all; and 30 corresponding game outcomes are recorded.
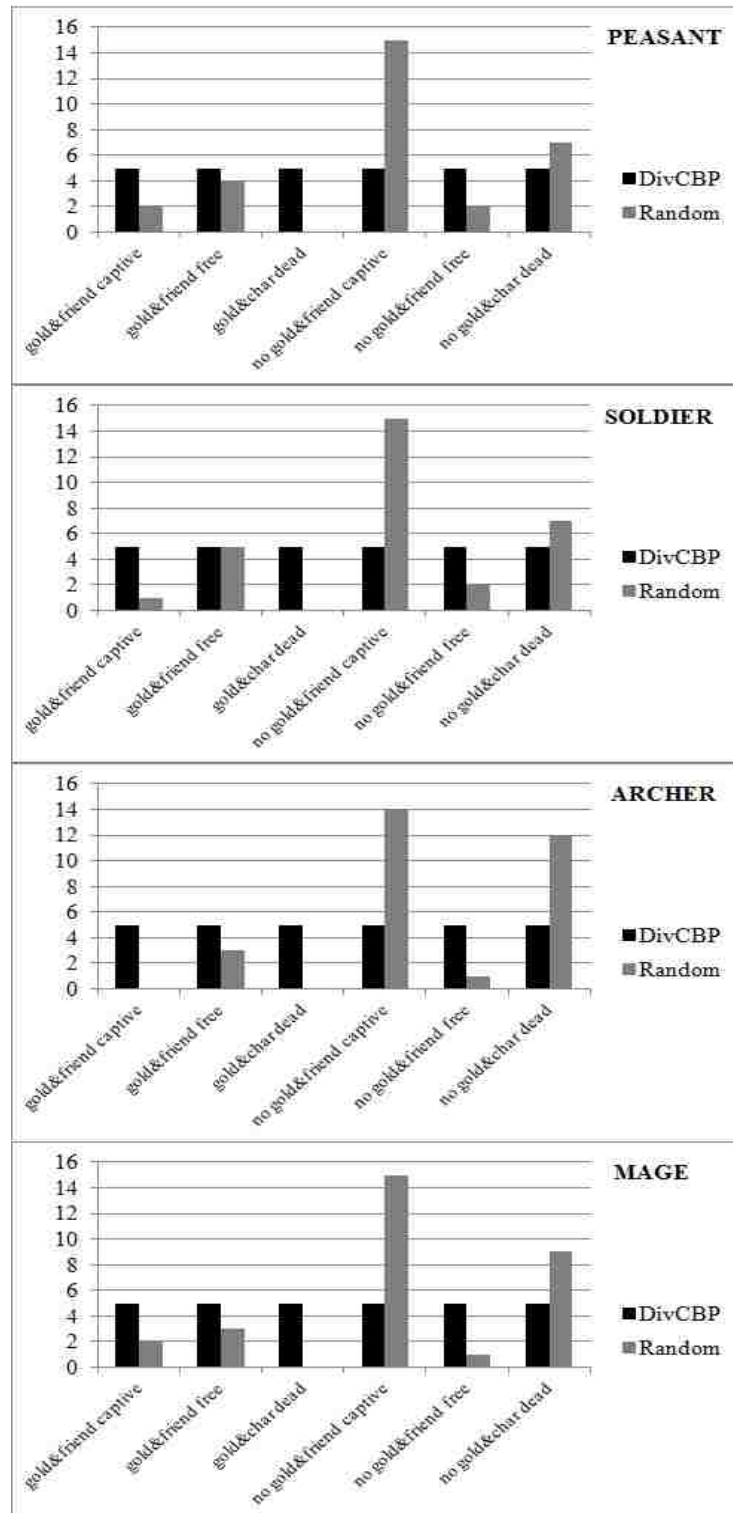
Figure 18. Number of outcomes of each type for each of the four unit categories

148

8.5.2 Evaluation Method

To evaluate behavior diversity, Shannon entropy is computed over the possible combinations of modifications to the game map caused by running the retrieved character plan.

The higher the entropy, the higher the uncertainty regarding the final map status. When this uncertainty is high, the generated plans are diverse in terms of the ways in which they affect the game environment when executed.

8.5.3 Experimental Results

In Figure 18, the bars represent the number of outcomes of each type for DivCBP and baseline Random selection (as indicated). The closer the bars corresponding to one method are to each other in terms of height, the more spread out the results are over the possible outcomes, indicating greater game-play variety. As can be seen, DivCBP results are consistently better spread out over the possible outcomes than Random results.

The entropy of DivCBP results is consistently greater than that of Random results (1,1,1,1 vs. 0.73; 0.71; 0.59; 0.69), indicating that, when using DivCBP, the uncertainty regarding the character type that will be chosen is greater; with random selection, certain character types tend to be favored over others.

This indicates that selection based on DivCBP more reliably produces diverse character types (the diversity of which is reflected in different in-game outcomes) than Random selection.

8.6   DivNDP and Policy-Based Character Variation

8.6.1   Experimental Setup

DivNDP is implemented using JavaFF (Coles *et al.*, 2008) as the heuristic planner employed by NDP.

The implementation of the NDP component of DivNDP is similar to the one described by Fu *et al.* (2011), including their two extensions, state reuse and goal alternative, which are shown to contribute to outperforming other strong-cyclic planners.

As in the work of Fu *et al.* (2011), and unlike in that of Kuter *et al.* (2008), who are restricted by the need to make NDP usable with any heuristic planner without modifying the planner itself, changes are made directly to JavaFF.

Sets of $k = 4$ diverse policies are generated. The value of α is set to 0.8, so as to encourage the generation of highly diverse policy sets.

To showcase the domain-independence of DivNDP, it is first tested on the synthetic nondeterministic planning domains Nondeterministic Blocksworld and Robot Navigation (with 7 doors), as described in Kuter *et al.* (2008) and Hogg,

Kuter, and Muñoz-Avila (2009). For these domains, the quantitative distance metric D_{PS} (Equation 14) is used.

Next, the main experimental evaluation is conducted on the Wargus domain. Actions in Wargus are nondeterministic (e.g. the success of an attack action is not guaranteed). Using nondeterministic planning makes it possible for the generated solution to succeed in the game even in the case of unexpected losses. For example, a policy might specify that a *soldier* should attack if it is alive, and that an *archer* should attack otherwise. Furthermore, the policy might indicate that a new *soldier* unit should be created if no such unit is available.

The game scenarios vary in terms of the elements on the map (e.g. villages, treasures) and the locations of these elements. The distance metrics are independent of the map (i.e. they do not specify any particular map coordinates). Units can move to various map locations, build villages, collect available treasure, and attack hostile and neutral locations.

Nondeterminism manifests itself in battle: the result of an attack action can be either success (the unit has killed the enemy without being killed itself) or failure (the unit was killed). A dead unit can be replaced by creating new units: this ensures that it is always possible for DivNDP to generate strong cyclic policies. In the initial state, units of all types are available.

The goal is to destroy all enemies on the map.

151

In Wargus, both **inflation-based** and **goal-achievement** qualitative distance are exemplified.

Goal-achievement distance is computed using Equation 24, which considers policies to be different if the types of units chosen to attack with when all units are still available are different. For example, the state of all units being alive and at the enemy camp might be associated with the action of a *soldier* attack in the first policy, and with the action of an *archer* attack in the second one. This reflects tactical variation.

In Equation 24, π and π' are policies, and *FirstAtt*(π) is the *type of attacking unit* (*archer*, *soldier*, *mage*, or *peasant*), while *d* is a domain-specific degree of distance between unit types (e.g. a *peasant* is considered more similar to a *soldier* than to a *mage* because the former two employ short-range attacks, while the latter employs long-range attacks).

$$D_{GA}(\pi, \pi') = \quad \begin{cases} 0, if\ FirstAtt(\pi) = FirstAtt(\pi') \\ d, 0 < d \leq 1, if\ FirstAtt(\pi) \neq FirstAtt(\pi') \end{cases} \quad (24)$$

Inflation-based distance (Equation 25) is used to simulate diversity in terms of character personality traits. In Equation 25, π and π' are policies, *CharTrait*(π) is the *character trait* reflected in policy π through small side-stories not essential to reaching the goal, while *d'* is a degree of distance between possible personality

152

traits. The personality traits represented are: (1) *ruthless* (attacks non-hostile village), (2) *hard-working* (builds extension to village), (3) *greedy* (collects treasure), (4) *indifferent* (does none of the above, focusing solely on achieving the goal).

$$D_{Personality}(\pi, \pi') =$$

$$\begin{cases} 0, if\ CharTrait(\pi) = CharTrait(\pi') \\ d', 0 < d \leq 1, if\ CharTrait(\pi) \neq CharTrait(\pi') \end{cases} \quad (25)$$

These two types of qualitative distance are compared to a quantitative distance metric. It will be shown that by using qualitative policy distance metrics, which encode meaningful, domain-specific differences between policies, the game environment is influenced in more diverse ways than by using quantitative policy distance metrics, which are domain-independent.

For quantitative distance (*RelD*$_{PS}$, Equation 18) the efficiency of DivNDP has been enhanced through a modification tailored to quantitative distance metrics like *RelD*$_{PS}$: the distance between the current partial plan and the set of previously-generated policies is computed cumulatively, and not completely recomputed at each step.

8.6.2 Evaluation Methods

To evaluate the diversity of the sets of policies generated with DivNDP on the synthetic domains, policy-set diversity (Equation 2) is used with the corresponding distance metric. Using the same metric for policy generation and evaluation is fair because, during policy generation, DivNDP compares partial plans to complete policies, whereas, in the evaluation, the diversity of the actual set of complete generated policies is computed.

For the Wargus domain, similarly to the evaluation in Chapter 8.5, Shannon entropy is computed over the possible combinations of modifications to the game map (e.g. treasure picked up, villages destroyed, units lost) caused by running the policy until the goal has been achieved. For goal-achievement distance, the map elements in the enemy-camp region of the map are considered, while, for inflation-based distance, map elements in the other regions are considered.

Note that all policies lead to the achievement of the goal. However, while all final states include the achieved goal, there is variation regarding the other facts in the final state: e.g. whether villages have been plundered or extended. None of these possible facts are considered more desirable than others a priori.

In addition to indicators of diversity, the average generated policy size is also reported. As Kuter *et al*. (2008) point out, while no good measure of quality is known for nondeterministic planning, policy size should be a good indicator of

quality (with smaller policies being preferable). As in the work of Fu *et al*. (2011), policy size is measured as number of state-action pairs.

8.6.3 Experimental Results

Synthetic Domains. Experimental results for Nondeterministic Blocksworld and Robot Navigation are presented in Figures 19 a) and b), respectively. Experiments were run on 100 problems per domain, divided into multiple sets by size (measured as number of blocks for Blocksworld, and number of objects that must be moved for Robot Navigation). As in the work of Hogg, Kuter, and Muñoz-Avila (2009), problems are of up to 8 blocks for Nondeterministic Blockworld, and of up to 5 objects for Robot Navigation. Each point on each of the two charts represents the average policy-set diversity (Equation 2, with $D = D_{PS}$) for 20 problems of the same size (each of them run 10 times).

Policy-set diversity values are, for the most part, very high. While the diversity is lower for the least difficult problems (which have small search spaces, not allowing for much variation), it increases quickly and remains above 0.5 for the majority of problems in both domains.

The average **policy size** of the diverse solution sets is only slightly higher than that of the first generated policy, indicating that quantitative diversity is not achieved by inflating policies unreasonably.

155
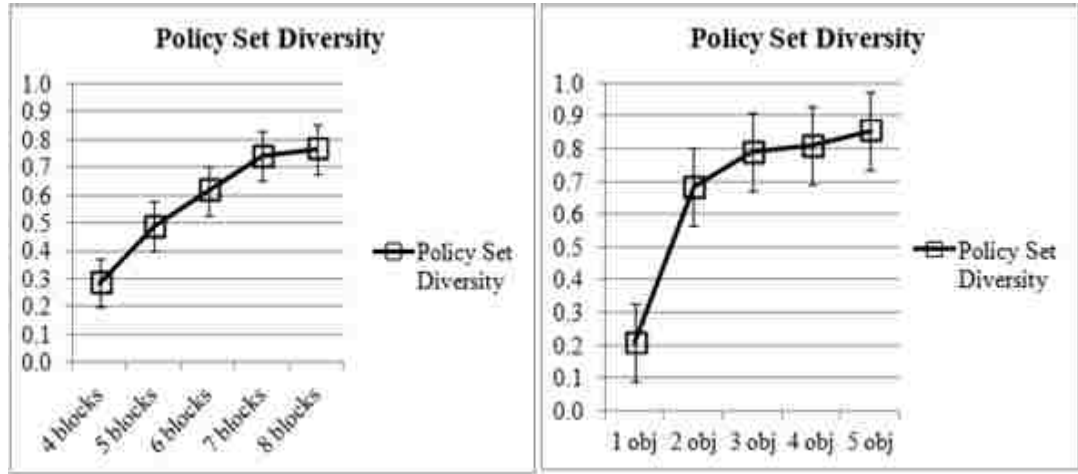
Figure 19. Policy-Set Diversity Results on the synthetic domains: (a) Nondeterministic Blocksworld, (b) Robot Navigation
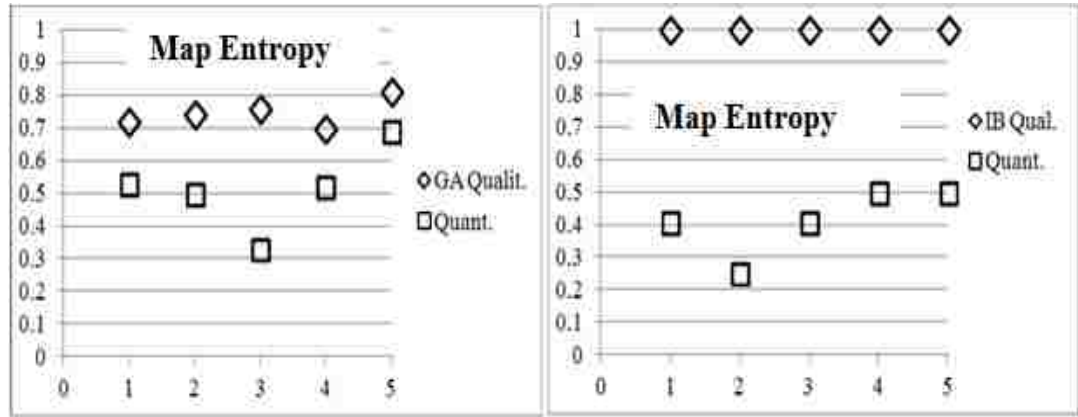
Figure 20. Map Entropy Results on the Wargus domain: Quantitative Distance vs. (a) Goal-Achievement, (b) Inflation-based Distance

Wargus Domain. Experimental results for the Wargus domain are presented in Figures 20 a) for goal-achievement qualitative distance and 20 b) for inflation-based qualitative distance. Each point on each chart is the average **entropy** value over multiple runs of generated policies for one game scenario. Each policy is run

in the game 25 times. Note that, due to the nondeterminism of the game, when running the same policy in the game multiple times, there can be different outcomes every time.

For both inflation-based and goal-achievement qualitative diversity, the entropy is consistently greater for qualitative diversity than for quantitative diversity.

Policy-set diversity (Equation 2, using the corresponding qualitative distance metric) is always maximal for both inflation-based and goal-achievement qualitative diversity.

Average **policy size** is slightly larger for quantitatively-diverse policies than for qualitatively-diverse policies. Quantitatively diverse policies are often inflated in a non-meaningful way, containing state-action pairs added solely for the purpose of increasing policy-set diversity.

9 Related Work

Diversity has diverse manifestations throughout Artificial Intelligence. They include solution diversity, goal diversity, search-space diversity, and problem formulation diversity. Diversity is created through specifically-designed techniques, using distance metrics or knowledge-intensive domain models; or implicitly, as a side effect of techniques primarily serving other purposes. It is pursued for its own sake, as an aid to exploring vast search spaces, and as an approach to fault-tolerance. Planning application domains which have been shown to benefit from diversity, or which diverse planning techniques have been applied to, include intrusion-detection (Boddy *et al.*, 2005; Roberts *et al.*, 2009), robotics (Lussier *et al.*, 2007), travel planning (Myers and Lee, 1999), and interactive storytelling (Guilherme da Silva, Ciarlini, and Siqueira, 2010). Outside planning, diversity appears in branches of Artificial Intelligence including case-based reasoning, general heuristic search, genetic algorithms, and constraint programming.

In the following subchapter (Chapter 9.1), I exemplify diversity in branches of Artificial Intelligence other than planning, most notably in case-based reasoning for analysis tasks. Then, I address related work in planning: first, I explore different approaches to comparing plans for purposes other than generating diverse sets of plans (Chapter 9.2); then, I provide an overview of alternative approaches to

diversity in planning (Chapter 9.3). Chapter 9.4 is dedicated to various relevant approaches to modeling characters and diversifying characters' behavior in computer games and other types of interactive fiction. Finally, I present an overview of comparative studies of first-principles and adaptation-based planning techniques (Chapter 9.5).

9.1   Diversity in Artificial Intelligence

9.1.1   Case-based Reasoning

Solution diversity has been explored extensively in case-based reasoning for analysis tasks, as defined in Chapter 2.3 (Smyth and McClave, 2001; Shimazu, 2001; McSherry, 2002; McGinty and Smyth, 2003; Bridge and Kelly, 2006).

Typically, case retrieval in case-based reasoning (see Chapter 2.3) is conducted based on similarity of the candidate-case problem to the new problem, or the estimated effort it would take to adapt the solution of the candidate case so that it fits the new problem (Aamodt and Plaza, 1994; Smyth and Keane, 1998). Smyth and McClave (2001) introduced the additional retrieval criterion of case diversity, citing recommender systems as the main application domain in which diversity would be useful.

Recommender systems match a user's need, be it clearly or formulated or not, to an entry from a library, most commonly describing a product or service offered by an e-commerce provider.

Content-based recommender systems retrieve cases containing descriptions of products or services (commonly represented as attribute-value pairs). Cases are assessed based on how well they seem to match a user's needs (expressed specifically through a query or inferred from item ratings in a user profile).

Collaborative recommender systems retrieve cases representing items which have appealed to individuals the profiles of which are assessed to be similar to the current user's (Bridge and Kelly, 2006). Actual item descriptions are not used.

Smyth and McClave (2001) propose an algorithm for increasing the diversity of retrieved sets of cases. Two variants of the algorithm are introduced: Greedy and Bounded Greedy.

The Greedy algorithm is the basis of Algorithm 7, and its general principles have already been explained (Chapter 4.2). Bounded Greedy is a variant of Greedy intended to be used when dealing with large search spaces, in which repeatedly computing relative diversity between the current candidate case and the set of previously-retrieved cases is costly.

Bounded Greedy consists of first ordering the cases based on similarity to the new problem then conducting Greedy retrieval only on a subset of cases ranked the highest.

Smyth and McClave define the diversity of a set of cases as the average dissimilarity of pairs of cases in the set, a definition which is used herein with the modification that plans, rather than cases, are subject to comparison.

160

Shimazu (2001) describes a different method for the retrieval of a diverse set of cases by a recommender system. The method consists of first retrieving a case that is maximally similar to the new problem, then retrieving a case that is maximally dissimilar to the first case, then retrieving a case that is maximally dissimilar to the first two cases. While retrieving a set of maximally diverse cases, this method takes no measures to preserve similarity to the new problem of the second and third retrieved cases.

McSherry (2002) proposes two algorithms which address the similarity and diversity trade-off: Similarity-Preserving Diversification, which creates diversity at no expense to similarity, and Similarity-Protected Diversification, which, while not keeping similarity intact, helps reduce the impact upon it.

The Similarity-Preserving Diversification algorithm consists of ranking cases based on their similarity to the new problem, then grouping them in "similarity layers" based on the value of the similarity between them and the new problem. All cases from all but the lowest similarity layers are retrieved. In addition, a subset of cases from the lowest similarity layer are selected: the cases which increase the diversity of the set of retrieved cases. If there is only one similarity layer, the case in it ranked highest based on diversity is retrieved, then the diversifying method is used on the remaining cases.

Similarity-Protected Diversification follows the same steps, but uses "similarity intervals" instead of similarity layers: the range of similarity values of

cases with regard to the new problem is divided into intervals of width α. Unlike a similarity layer, a similarity interval does not necessarily contain only cases which are equally similar to the new problem. The leftmost (highest-similarity) interval contains cases such that the similarity between them and the new problem is greater than 1 - α.

Of the methods listed above, Shimazu's (2001) is concerned primarily with diversity, while McSherry's (2002) Similarity-Preserving Diversification focuses on similarity. Greedy and Bounded Greedy (Smyth and McClave, 2001) and Similarity-Protected Diversification (McSherry, 2002) are the approaches most concerned with providing balance between the two considerations (however, Similarity-Protected Diversification supports similarity more strongly than the algorithms of Smyth and McClave, 2001).

Out of these approaches, that of Smyth and McClave is clearly the most similar to the work presented herein: it uses a composite criterion incorporating problem similarity and diversity to select all but the first case in the retrieved set. The algorithm does not inherently favor either similarity or diversity: the complementary weights of the two criteria are adjustable. Algorithm 7 is a variant of the Greedy Retrieval algorithm of Smyth and McClave.

The Bounded Greedy algorithm is also used by Bridge and Kelly (2006) and McGinty and Smyth (2003).

Bridge and Kelly (2006) use Bounded Greedy to demonstrate retrieval of diverse sets of recommendations for collaborative (rather than content-based) conversational recommender systems, using various types of distance metrics.

McGinty and Smyth (2003) describe Adaptive Selection, a diverse-case retrieval method intended for **conversational recommender systems** (Aha, Breslow, and Muñoz-Avila, 2000). Conversational recommender systems do not produce only a single set of retrieved cases, but conduct a prolonged dialogue with the user (as a human salesperson might), proposing sets of products and refining their recommendations based on the user's feedback.

Adaptive selection chooses dynamically whether or not to include diversity as a retrieval criterion in each recommendation cycle.

If the user picks an item that was introduced in the latest recommendation cycle, this is judged to mean that progress has been made toward the user's ideal match, hence the next cycle will consist solely of the items which are more similar to the selected one (i.e. retrieval occurs based only on the similarity criterion).

If the user picks the item that was carried over from a previous recommendation cycle (the item which was preferred by the user in the previous cycle), this is judged to mean that no progress has been made during the current cycle, hence the user would benefit from being presented with a more diverse set of choices. Therefore, the next set of recommended items is retrieved using Bounded Greedy (Smyth and McClave, 2001).

While no process akin to adaptive selection is being used herein, this technique could serve as inspiration for future work handling the efficiency burden of generating diverse plans using first-principles planners (see Chapter 10.1 for details), or addressing diversity in mixed-initiative planning, wherein the human collaborator may benefit more from diverse options at certain times, and from similar ones at others.

Bridge and Ferguson (2002) enhance diversity through order-based retrieval, a method which ranks cases based on ordering operators reflecting more complex orderings than that used for traditional similarity-based retrieval. Some such ordering operators are shown to be conducive to diversity without necessarily incorporating distance measures (although the possibility of defining distance-based ordering operators is mentioned). The fact that diversity arises as a side-effect of order-based retrieval, without diversity considerations being specifically imposed is a fundamental difference from the approach taken herein, which specifically reinforces diversity.

Diversity concerns appear in case-based reasoning literature outside recommender systems also. Zhang, Coenen, and Leng (2002) use diversity-enhancing techniques at the retrieval stage of a case-based reasoning diagnosis tool. Benefits are expected from retrieving cases which do not reflect the same faults.

It must be stressed that most work on diversity in case-based reasoning was conducted in the context of analysis, rather than synthesis tasks (see Chapter 2.3 for

an explanation of the difference between the two). My work addresses diversity in planning, which is a synthesis task and requires new diverse structures (plans and policies) to be created, not just identified in a set of preexisting items. Furthermore, in analysis tasks, both the similarity and diversity retrieval criteria are applied to the problem component of the case, while, in case-based planning, the diversity criterion is applied to the solution plans. Plans, which are complex structures with an arbitrary number of actions, each with an arbitrary number of parameters, can be considerably more difficult to compare than typical analysis-task case problems (e.g. recommender-system item descriptions consisting of a fixed number of attributes).

Unlike the authors of the related work presented above, Díaz-Agudo *et al*. (2008) address "originality-driven" tasks, which are synthesis (rather than analysis) tasks where the objective is to produce a new artifact that is significantly different from the items in the case base, thus illustrating creativity (storylines are the featured example). However, unlike in my work, the authors do not specifically introduce distance criteria in the solution generation process, but note that originality occurs as a "side-effect" of other factors: the size of the solution space and the need to enforce solution consistency. As a future work direction, they mention the possibility of introducing distance-metric-based evaluation at the reuse (adaptation) stage of the case-based reasoning cycle, rather than at the retrieval stage (as herein).

Another important difference between related work on diversity in case-based reasoning and my work is that qualitative diversity is not considered therein. The systems described above create quantitative diversity: there are no domain-specific degrees of distance between solution elements (quantitative distance metrics used include Hamming Distance, as used by Bridge and Kelly, 2006).

9.1.2   Other Branches of Artificial Intelligence

Search Problems. Beam Search (Newell, 1978) is a variant of heuristic search in which only a subset of the generated candidate states are retained during each step of the search, thus providing a feasible approach to handling very large search spaces. This subset of retained cases is called the **beam**. As a number of candidate states are pruned, this type of search is incomplete: it does not guarantee that a solution will be found. The danger is that candidate states which might have lead to a good solution will be subjected to pruning, and convergence to local maxima will prevent the algorithm from finding global optima.

Shell, Hernandez Rubio, and Quiroga Barro (1994) propose a Beam Search variant in which the diversity of candidate states maintained in the beam is specifically enforced. The reasoning behind this is that more diverse candidates will cover a larger area of the search space, reducing the likelihood of convergence to local maxima.

While the motivation that diverse candidates provide a better sample of the search space is similar to the one invoked herein, the main difference between their work and mine (aside from the fact that their work addresses general search, rather than search for the purposes of planning) is that they do not pursue diversity for its own sake (by aiming to produce a set of diverse final solutions), but use it as a means to achieving the objective of finding a solution in a very large search space that it is costly/unfeasible to browse exhaustively. The method of selecting diverse sets of nodes is similar to the one presented herein in the sense that a composite criterion taking into account both adequacy and diversity is used to assess candidate states. The purpose of the assessment is to determine whether a state will be pruned or maintained as part of the search space. An initial set of states is selected based on adequacy only (in order not to forego the states which the regular heuristic evaluates as being most promising), then another subset of states, selected based on the composite criterion, is added to the previous one. The candidate comparison criterion used is quantitative (although the possibility of using qualitative criteria is mentioned).

Multiobjective/Multicriteria Heuristic Search (Stewart and White, 1991; Dasgupta *et al.*, 1999; Mandow and Pérez de la Cruz, 2003) is a variant of heuristic search conducted in a search graph in which arcs (representing transitions between states) have multiple cost values associated to them, each cost value corresponding to one different objective that must be pursued in the problem domain (e.g. risk,

transportation cost, etc.). The solution to such a problem is a path in the search graph that is Pareto-optimal with regard to a set of objectives, or the set of nondominated paths from the initial state to goal states. A path is nondominated if no other path is at least as good as it in terms of all objectives and better than it in terms of at least one objective. MOA* (Stewart and White, 1991; Dasgupta *et al*., 1999; Mandow and Pérez de la Cruz, 2003) a variant of the A* heuristic search algorithm, is used to solve multiobjective search problems.

Unlike in my work, the objective is not diversity for its own sake, but optimizing multiple objectives, through one or more solutions. I address planning problems which are not formulated as optimization problems.

Genetic Algorithms (DeJong, 1975) are search algorithms modeled on the processes underlying biological evolution: candidate states called "genotypes" are generated through operators named, after the natural processes they mimic, "crossover" and "mutation". A "fitness function" (the counterpart of a heuristic evaluation function in regular search) is used to evaluate genotypes and select the ones most likely to "reproduce" (i.e. be expanded).

Diversity was introduced in genetic algorithms in order to alleviate a difficulty they have in common with other types of search algorithms: convergence to local maxima in the search space. Much like gene pool variety is of inestimable value in biology, diversity has been shown to be valuable in genetic algorithms. Steps taken both to reduce the similarity of genotypes (DeJong, 1975) and to actively enforce

their variation through diversity criteria (Mauldin, 1984) have resulted in performance gains.

As in previously-presented search types, the main difference, in objective, between their work and mine is that they do not pursue diversity for its own sake, by seeking to produce diverse sets of solutions, but as a means to alleviating the issue of convergence to local maxima, so as to improve search performance.

Constraint Programming. Hebrard *et al*. (2005) address the issue of generating diverse and similar solutions in constraint programming (Rossi, Van Beek, and Walsh, 2006) which is used, among others, to solve constraint satisfaction problems. Hebrard *et al*. define a series of similarity/diversity-related tasks and propose methods to solve them. The diversity-related tasks are: finding a solution such that the distance between it and another given solution surpasses a certain threshold, finding the set of k maximally diverse solutions to a problem, and finding a set of solutions such that the distance between each of them and another given solution surpasses a specified threshold value.

Herein, the objective is to find a set of diverse solutions, but maximal diversity is not pursued because of the prohibitive size of the search spaces in planning problems, particularly nondeterministic planning problems. Also, in my work, there are no guarantees of minimal distance in between solutions.

Goal Variation in Agent-Based Systems. Goal-Driven Autonomy (Molineaux, Klenk, and Aha, 2010) is an extension to on-line planning (i.e. planning that is

interleaved with execution, rather than being completed prior to it) which allows artificial intelligent agents to generate, reason about, and, if necessary, modify the goals they are to pursue. Goal modifications are undertaken as a result of identifying discrepancies (i.e. conditions, brought about by unexpected events, which differ from those assumed when the plan was initially generated) in the environment, which the agent monitors. For example, in a combat-based computer game domain, in response to an unexpected enemy attack, the current goal might change from harvesting enough resources to build a new town hall to taking urgent defensive measures.

Note the differences between this approach and diverse nondeterministic planning as addressed herein: in nondeterministic planning, actions have multiple possible outcomes and multiple goals, but the set of goals remains unchanged, whereas, in Goal-Driven Autonomy, the goals themselves can change. In the former case, it is the solutions which are diverse; in the latter case, it is the goals.

9.2   Plan Comparison

A number of authors have previously addressed the problem of comparing plans for purposes other than generating diverse plans. Fox *et al*. (2006) and Felner *et al*. (2007) use plan comparison for the purpose of generating plans which are similar, rather than diverse. Possible reasons for this endeavor include commitments to other agents in multi-agent requirements and the potential preference of human

planners in a mixed-initiative context for plans which appear more familiar to them (van der Krogt and de Weerdt, 2005).

Fox *et al*. (2006) use plan comparison for the purpose of minimizing, rather than maximizing, the difference between a source plan and a target plan produced through plan repair.

Plain repair is used when execution conditions differ from those expected at planning time, or when goals change during execution. It consists of adapting a previously-generated **source plan** into a **target plan** addressing the identified discrepancies in the environment conditions or changes in goals.

Producing stable repaired plans is particularly important in high-risk domains, in which safety is foremost. Fox *et al*. use a quantitative similarity metric computed by adding the number of actions in the first plan but not the second to the number of actions in the second plan but not the first (Equation 4 is based on this metric). Planning is conducted using a modified version of the LPG (Gerevini, Saetti, and Serina, 2003) planner.

Felner *et al*. (2007) also address the situation in which it is necessary to find a new plan that is similar to a previously-generated plan which can no longer be used due to discrepancies between assumptions made at planning time and the actual environment conditions. The distance metrics they use are also quantitative but, unlike the metric used by Fox *et al*. (2006), they are not concerned with the actions plans consist of, but with the intermediary states assumed to be brought about by

the actions: the plans are represented as paths in graphs in which the states are nodes. Taking as input a graph-based representation of the planning domain, a previously-generated plan represented as a path in the domain graph, and a plan distance metric, their algorithm uses best-first search with various heuristics to generate plans that are similar to the preexisting input plan. The work of Felner *et al.* is similar to mine in that it is flexible with regard to the distance metric used to compare plans: this distance metric is part of the input.

While pursuing the opposite goal (solution similarity rather than diversity), the approaches of Fox *et al.* (2006) and Felner *et al.* (2007) are similar to the one presented herein in that they modify regular heuristic-search methods to include plan-comparison criteria.

Myers (2005) conducts qualitative plan comparison through the use of a domain metatheory. A **domain metatheory** is an extended description of the planning domain in terms of high-level attributes (such as features of operators, e.g. whether a transportation method is aerial or by ground), supplementing the standard domain model.

The qualitative comparison takes place within the HTN planning paradigm. The objective is to summarize plans and highlight their similarities and differences in a way that is immediately meaningful and useful to the human users they are presented to. This time, differences are identified not by using distance metrics focusing on the low-level components of plans (e.g. actions and states), but on the

basis of high-level characteristics described in the metatheory, combined with the extensive task descriptions required for HTN planning. These differentiation criteria include the roles played by various objects in the context of tasks, and high-level characteristics of tasks (such as difficulty, cost, and risk). Comparison is conducted both within pairs of plans and over entire sets of plans. An example of the latter is identifying a plan which has a characteristic unique within the analyzed set, e.g. a very high risk associated to it, when the risk level of all other plans is at most moderate. Another example is identifying two maximally diverse plans within a set of plans. Unlike in my work, actual diverse plan generation is not conducted here: this is merely a question of comparing already-available plans.

9.3   Diversity in Planning

Tate, Dalton, and Levine (1998) address qualitative solution diversity in a mixed-initiative planning setting. Their method involves populating a "course of action" matrix with varied potential solutions to a problem (referred to as "options"). The matrix is used to compare and evaluate these options. A great part of the cognitive load, including that associated with creating diversity, falls upon the human planners (with the AI planning system serving as aid, rather than principal planner), making this approach vastly different from those proposed herein.

In generative planning, quantitative plan diversity has been explored by Srivastava *et al.* (2007) and Nguyen *et al.* (2012). They describe a diversity-aware

modification of the LPG (Gerevini, Saetti, and Serina, 2003) heuristic planner. This diverse planning system is subsumed by the general diverse planning framework presented here, i.e. it iteratively generates diverse plans using a composite evaluation criterion adding to the regular LPG heuristic function a diversity component. The distance metric used by this diverse planner is similar to Equation 4. The main differences between their work and mine are that they only generate quantitatively diverse plans, which, as has been shown, may not be meaningfully diverse; they only address deterministic planning problems, and they only evaluate their work on synthetic domains, while the systems presented herein are evaluated on the Wargus real-time strategy game domain as well synthetic domains. Srivastava *et al.* and Nguyen *et al.* also present a diverse version of a constraint-satisfaction planning system, demonstrated with multiple distance metrics.

In addition, Nguyen *et al.* (2012) also address the problem of generating sets of plans maximizing multiple objectives (plan makespan and execution cost) in temporal-planning-with-preferences settings in which the user preferences are only partially expressed. My work is concerned neither with planning with preferences nor with generating sets of plans that optimize objective functions.

A method for qualitative-diversity-aware plan generation has been proposed by Myers and Lee (1999). Their approach uses a domain metatheory (of the same type as the one used by Myers, 2005, for plan comparison, as described in Chapter 9.2) to divide the search space into regions representing sets of qualitatively different

plan characteristics, then "biases" the planner towards making choices resulting in different such regions being represented in the generated set of plans. Their method is demonstrated in the context of HTN planning. Unlike the method of Myers and Lee, the qualitative solution generation approach (or, rather, the flexible approach which supports both qualitative and quantitative diversity) presented herein does not rely upon a metatheory: it only requires a qualitative distance metric, in addition to the basic planning domain description.

Eiter *et al.* (2011) address solution similarity and diversity in the context of answer-set programming problems. Answer-set programming (Gelfond and Lifschitz, 1991) is a declarative programming paradigm similar to propositional-satisfiability solving. Eiter *et al.* define similarity/diversity-related tasks similar to those described by Hebrard *et al.* (2005) in the context of constraint-satisfaction problems (see Chapter 9.1.2). One of the application domains they target is deterministic planning: planning problems are encoded as answer-set programming problems and solved with proposed diversity-aware answer-set programming techniques (involving, among others, reformulating problem descriptions so as to include the diversity requirements). The distance metric used to compare plans is quantitative and the planning domain used for evaluation is synthetic (Blocksworld).

Lussier *et al.* (2007) propose an application for diversity as a means of achieving fault-tolerance in the context of autonomous systems (robots). Sets of

diverse plans are used to increase the robustness of such systems. An exemplified diversity-based recovery mechanism is that of reacting to error detection by switching to a variant of the planner using a different variant of the knowledge model, in the expectation that the change will likely lead to generating a plan sufficiently different that the error will not be encountered. Plan diversity, therefore, is not created by using a diversity-aware modification of a planning technique (as herein), but through variations in the knowledge presented to the planner.

In *probabilistic* nondeterministic planning, Bryce *et al.* (2007) generate multi-option plans, using a modified version of the LAO* algorithm (Hansen and Zilberstein, 2001). Multi-option plans associate to a state multiple actions, each optimizing the planning objectives in a different way: a multi-option plan is a representation of the set of plans that is Pareto-optimal with regard to the set of objectives. This endeavor is related to multiobjective heuristic search (Section 9.1.2). I address *non-probabilistic* nondeterministic planning, in which it is not assumed that goals are encoded as objective functions, hence planning tasks are not handled as optimization tasks, and one cannot create diversity by generating sets of solutions optimizing multiple objectives. In the work of Bryce *et al.*, diversity arises from optimizing multiple objectives; in my work, solution-set diversity is, in itself, the objective.

Sroka and Long (2012) propose a research direction also concerned with generating the Pareto set of plans optimizing multiple objectives: this time, for deterministic planning problems. Unlike in the work presented herein, they assume the availability of an objective function (expressing plan-quality-related preferences) as part of the problem description, and separate from the problem goal. They propose plan comparison to be conducted using this objective function. Diverse plan sets are generated by running the planning system on the same problem multiple times, each time with a slightly different objective function (the weights assigned to the objectives in the function vary). The distance metrics used in my work (unlike the objective functions in theirs), remain the same over the multiple runs of the planner, and it is the planner itself that has been modified for diversity.

Guilherme da Silva, Ciarlini, and Siqueira (2010) use nondeterministic planning techniques to generate storylines for interactive storytelling domains. One of their main justifications for using nondeterministic, rather than deterministic, planning techniques is the need to create diverse storylines. Storyline diversity is not enforced specifically, but considered to be an implicit effect of using nondeterministic planning, which allows for multiple possible outcomes for each action.

As part of the motivation for diversity in planning, I have mentioned the usefulness of diverse plans reflecting different possible attack strategies in

intrusion detection. This need was initially pointed out by Boddy *et al.* (2005). Their approach to diversity was repeatedly tweaking the domain and problem descriptions themselves so as to reflect varied vulnerabilities. Hence, the generated plans were not diverse solutions to the same problem, but solutions to different problems.

Roberts *et al.* (2009) address diverse plan generation specifically. They describe ITA*, a diverse planner based on the state-of-the art heuristic-search planner LAMA (Richter and Westphal, 2010). It does not use modified planner heuristics, but keeps a list of state-operator pairs that have been included in previously-generated plans, and prioritizes state-operator pairs which are not included in this list. This method is inherently quantitative: any distinct state-action pairs are maximally distant; there are no domain-specific degrees of distance between pairs. While motivated by needs characteristic of the intrusion-detection field, ITA* is domain-independent. It is evaluated experimentally using, as baseline, DivA*, a variant of the DivFF planner presented in Chapter 4.1, with the quantitative distance metric. While the general principles of DivFF as well as the quantitative distance metric used by it remain the same, various changes are being made in order to maintain fairness in the experimental comparison against ITA*: the most notable change is the use of A* heuristic search rather than the heuristic search method specific to FF. The authors conclude that the two diverse planning methods (their own and the one based on diverse DivFF) can be seen as

complementary, with various relative strengths and weaknesses. The comparison is conducted solely using quantitative diversity, as ITA* is specifically tailored towards quantitative diversity.

9.4   Game Character Modeling and Diversity

Complex game character modeling has been explored to such a great extent that, rather than attempt exhaustiveness, I will cite several illustrative examples. Non-player characters (NPCs) have been modeled in terms of personality traits (McCoy *et al*., 2010), emotional states (Cavazza and Charles, 2005; Strong *et al*., 2007; Cavazza *et al*., 2009), relationships (Cavazza and Charles, 2005; McCoy *et al*., 2010; Thue *et al*., 2010), and needs (McCoy *et al*., 2010; Paul *et al*., 2010). In the related field of behavioral robotics, emotions are modeled by Arkin *et al*. (2003).

Substantial work deals with the complexities of generating interesting and realistic language/dialogue that is character and situation-appropriate (Cavazza and Charles, 2005; Strong *et al*., 2007; Lin and Walker, 2011).

Another significant area of focus is NPC behavior in the context of storyline consistency and variety (Cavazza, Charles, and Mead, 2002; Porteous, Cavazza, and Charles, 2010a; McCoy *et al*., 2010; Thue *et al*., 2010). Several of these approaches to character modeling use HTN planning (Cavazza, Charles, and Mead, 2002; Cavazza and Charles, 2005; Paul *et al*., 2010) or other types of task-

decomposition-based planning (Porteous, Cavazza, and Charles, 2010a) to control storyline development and character behavior.

Learning character models from human or human-like behavior has been used as an approach to overcoming the knowledge-acquisition difficulties (Lin and Walker, 2011; Chang *et al.*, 2011).

In contrast to the above-mentioned work, the approach taken herein is creating NPC behavior diversity (based on plan/policy diversity) that simulates personality diversity, rather than modeling NPC personality which is then reflected in varied behavior.

Szita, Ponsen, and Spronck (2009) study diversity in the context of adaptive game AI based on reinforcement learning. Their motivation for this endeavor is that learning-based adaptive AI systems generally converge to a small set of tactics which prove effective against opponents but lack diversity: thus, the player may end up repeatedly facing NPCs executing roughly the same strategies. The authors mention NPC diversity as a significant factor in rendering game AI engaging, and they set out to create adaptive game AI exhibiting behavior that is both effective and diverse.

Their game characters are controlled by rule-based scripts, while, in my work, game characters are controlled by plans generated through AI planning techniques of which learning is not a component. In their work, the fitness function evaluating the learned scripts is modified so as to reward script diversity: this has similarities

to the use of the composite evaluation criterion including a diversity component in my work.

Goal-oriented action planning (Orkin, 2003) consists of endowing game characters with the ability to conduct real-time planning in order to achieve their goals. This makes these characters better adaptable to changeable environment conditions than characters acting out hardcoded behavior. It also relieves the game designer of the effort of encoding behavior scripts for each conceivable situation that the character might encounter. While not specifically using plan-diversity-enhancing techniques, Orkin attains character diversity as a welcome side-effect of allowing his characters the autonomy of pursuing goals through variable plans, rather than acting according to a hardcoded script.

Finally, I will list several more authors who use planning techniques in computer games, without diversity necessarily being a central factor (although, as seen above, the use of planning techniques in games can serendipitously lead to game diversity). Case-based planning for game environments has been demonstrated, among others, by Fairclough (2004) and Ontañón *et al*. (2010). Real-time planning has been demonstrated in a game environment by Bartheye and Jacopin (2009). While I do not address real-time planning herein, the presented character-diversity techniques could conceivably be applied in that context as well.

9.5 Comparative Studies of First-Principles Planning and Adaptation-based Planning

While this is the first work comparing first-principles planning and adaptation-based planning from the point of view of the diversity of the generated solutions, there have been numerous studies comparing the two approaches based on other criteria. First of all, it should be noted that case-based planning is not the only adaptation-based planning approach. By adaptation-based planning, I refer to any approach to planning which relies on information gathered in previous planning processes (be it the actual solutions themselves or information on how they were generated) to create solutions to new problems. Plan repair (as described in Chapter 9.2) is an example of adaptation-based planning.

In terms of planning efficiency, advantages of adaptation-based planning over planning from scratch have been demonstrated, among others, by Veloso (1994), Gerevini and Serina (2000, 2010), and van der Krogt and de Weerdt (2005), and proved formally by Au, Muñoz-Avila, and Nau (2002), and Kuchibatla and Muñoz-Avila (2006). Fox *et al.* (2006) demonstrate advantages of adapting available plans (over replanning from scratch) in terms of plan stability, which is a measure of how many actions in the source plan are revised; the fewer revised actions, the more stable the planning algorithm.

Initially, Nebel and Koehler (1995) showed, through complexity analysis, that certain formulations of the plan adaptation problem are more difficult than first-

principles planning. If this were true of all variants of the problem, it would call into question the very usefulness of the plan-adaptation approach to planning. However, experimental results seemed to be repeatedly contradicting this conclusion by showing that adapting previously-generated plans can, in practice, lead to increases in efficiency over planning from scratch. The explanation (Au, Muñoz-Avila, and Nau, 2002) is that the assessment of Nebel and Koehler referred to a very restrictive formulation of the plan-adaptation problem (called *conservative*). This formulation is based on the assumption that the modifications made to the initial solution in order to obtain a new solution are minimal (i.e. it is guaranteed that as much as possible from the preexisting solution will be reused). As pointed out by Au, Muñoz-Avila, and Nau (2002), actual implemented adaptation-based planning systems are not conservative.

That efficiency gains can, in practice, be attained through adaptation-based planning approaches has been shown repeatedly.

Veloso (1994) describes the planning system PRODIGY/ANALOGY, which combines derivational-analogy case-based planning with first-principles planning techniques. The adaptation-based enhancements reusing information from previous planning processes are shown to produce an increase in efficiency over the base first-principles planner.

Gerevini and Serina (2000, 2010) propose a plan repair method based on "replanning windows". A replanning window is used to isolate a portion of the

source plan that contains inconsistencies with regard to the new planning requirements, e.g. unsatisfied preconditions or unfulfilled new goals. Each replanning window is treated as a separate planning problem and solved using first-principles planning techniques. The goal of each such problem is to achieve either the preconditions of the action immediately following its corresponding window, or the actual goal of the repaired plan. After being generated, the solution plans for the problems corresponding to the replanning windows are inserted at the appropriate position in the source plan, thus obtaining the adapted plan. This adaptation-based approach is shown to be considerably faster than planning from scratch.

van der Krogt and de Weerdt (2005) propose an approach to plan repair which uses heuristic-based planning techniques. In a manner reminiscent of transformational adaptation in case-based planning, plans are repaired by removing and adding actions. Removal and addition is achieved through heuristic planning, with heuristic functions being used to assess the value of a removal or an addition. This approach to plan repair is experimentally found to be faster than planning from scratch in most instances.

In addition to work like that described above, which shows the advantages of plan adaptation through experimental evaluation of plan-adaptation-based systems, formal studies have also been conducted to show the same through complexity analysis.

Au, Muñoz-Avila, and Nau (2002) show that derivational-analogy case-based planning is not conducted under the conservative assumption of Nebel and Koehler (1995) and that it actually is, in the worst case, as difficult as first-principles planning. They also show that the search space for derivational-analogy systems is not larger than that of first-principles classical planners, and, in certain situations, it can be exponentially smaller. To conduct this study, they first describe a general framework for derivational analogy, which is then used alongside a similar preexisting framework for classical planning (Kambhampati and Srivastava, 1995) to compare the two planning approaches.

A similar result, this time in the context of transformational-analogy case-based planning systems, was published by Kuchibatla and Muñoz-Avila (2006). Similarly to Au, Muñoz-Avila, and Nau (2002), they start by defining a general framework for transformational-analogy case-based planning, then, using this framework, they conduct an analysis showing that transformational-analogy is not conservative: hence, the complexity analysis of Nebel and Kohler (1995) does not apply to it.

While the previously-described comparative studies focus on planning efficiency and related concerns, Fox *et al*. (2006) demonstrate advantages of adapting available plans over planning from scratch in terms of plan stability.

Stability is defined as a measure of the difference between a source plan and a target plan. It is presented as a planning-technique evaluation metric for problem

instances in which one is forced to make modifications to a preexisting source plan, but there is a preference toward target plans which are as similar as possible to the source one. As previously explained, the authors implement a plan-repair technique using a modified version of the heuristic-search planner LPG (Gerevini, Saetti, & Serina 2003). It is shown that the proposed form of plan repair is characterized by higher stability than creating new plans from scratch using the regular version of LPG. Advantages in terms of planning time and solution quality (with the adaptation of high-quality source plans being likely to result in high-quality target plans) are also shown.

10 Conclusion and Future Work

This work presents a general framework for diversity-aware Artificial Intelligence Planning.

The framework is based on **iterative generation of diverse solutions using distance metrics**. The objective that this framework is intended to fulfill is that of generating multiple diverse solutions to the same planning problem. This is achieved by repeatedly generating solutions based on a composite candidate solution evaluation criterion (Equation 1, repeated below for convenience). This evaluation criterion takes into account both how promising the candidate solution looks in its own right (*SolAdequacy*) and how high its potential for contributing to the diversity of the final solution-set is judged to be (*RelDiv*). At the basis of this estimation of diversity are **solution distance metrics**, measures of the dissimilarity between two solutions.

$$EvalCrit(\pi, \Pi) = \alpha SolAdequacy(\pi) + (1 - \alpha)RelDiv(\pi, \Pi) \qquad (1)$$

It has been shown that distance metrics can be **quantitative** or **qualitative**.

Quantitative distance metrics are domain-independent, with a typical such metric being computed as the number of plan elements (e.g., actions) which appear

in strictly one of the compared plans. Quantitatively-diverse solutions are not guaranteed to be meaningfully diverse.

Qualitative distance metrics incorporate domain-specific knowledge, which they use to compute the degree of meaningful dissimilarity between solutions.

In practice, implementations of the general framework do not have to be built from scratch, as they can conveniently make use of regular, non-diverse planning systems. These systems are modified so as to use diversity considerations at planning time (i.e. the relative diversity component is added to their candidate-solution-evaluation criterion). The solution-adequacy component of the composite evaluation criterion is provided by the regular planner itself.

This approach is taken herein, when using the general framework as the basis for three diversity-aware planning systems: DivFF (a diverse heuristic-search planner for deterministic planning problems), DivCBP (a diverse case-based planner for deterministic planning problems), and DivNDP (a diverse first-principles heuristic-search planner for nondeterministic planning problems).

The three systems represent three different planning-technique/planning-problem-type combinations. The planning techniques are heuristic-search planning and case-based planning. The problem types are deterministic planning problems (wherein it is assumed that any action may have only one given outcome) and nondeterministic planning problems (defined under the assumption that actions may have multiple possible outcomes).

DivFF, DivCBP, and DivFF are all flexible with regard to the type of distance metric they use to compare solutions, and have been tested with both quantitative and qualitative solution distance metrics.

In terms of novelty, DivCBP is the first diverse case-based planning system; DivFF is the first diverse heuristic-search planning system demonstrated with both quantitative and qualitative distance metrics, and, while a formulation of the diversity task different from the one pursued herein has been addressed in probabilistic nondeterministic planning, DivNDP is the first *non-probabilistic* nondeterministic planner capable of generating diverse sets of solutions. All systems are the first of their kind tested by running the solutions in the environment they are intended for (a real-time strategy game).

A comparison between DivFF and DivCBP (the two systems are comparable because they both generate solutions to deterministic planning problems) was conducted, in order to determine whether any one of them is better able to generate diverse sets of plans and, if so, why. This comparison between DivCBP and DivFF contributes to the body of comparative studies between first-principles and adaptation-based planning systems.

Potential applications for diverse-plan/policy-generating systems include, as shown in related work, intrusion-detection, fault-tolerance, and computer games and other forms of interactive storytelling (Boddy *et al.*, 2005; Lussier *et al.*, 2007; Guilherme da Silva, Ciarlini, and Siqueira, 2010).

A diverse-plan/policy application demonstrated herein is that of making non-player characters in a computer game not designed with character variety in mind appear diverse in terms of their personality traits. Character behavior can be represented in terms of plans or policies (Orkin, 2003), hence plan/policy diversity can be at the basis of character diversity. This is demonstrated both in deterministic planning, using DivCBP, and in nondeterministic planning, using DivNDP. Plans and policies representing character behavior simulating personality variety are generated using solution-distance metrics defined so as to detect indicators of different personality traits.

The experimental evaluation was conducted not only on synthetic planning domains, but also on a planning domain based on the real-time strategy game Wargus. DivFF, DivCBP, and DivNDP were all tested both with quantitative and with qualitative distance metrics, and assessed by running the solutions they generate in the game environment they were intended for, and observing behavior and results thus obtained. This is an important contribution, as, in related work on solution diversity in planning, diversity is assessed by analyzing the plans themselves, without actually running these plans (Myers and Lee, 1999; Srivastava *et al*., 2007; Nguyen *et al*, 2012; Eiter *et al*., 2011). The results of the experimental evaluation are summarized below.

Qualitatively-diverse solutions always produce markedly more diverse behavior, when run in the game environment, than quantitatively-diverse solutions.

Moreover, quantitatively-diverse plans/policies are often inflated with actions not necessary for reaching the goal, added solely for the purpose of increasing their distance from one another, thus making the solution set more diverse. On the other hand, inflation is not always counter-productive: it was shown how qualitative distance metrics can be used to encourage meaningful (rather than random) inflation that makes a policy representing a storyline more eventful.

DivCBP generates sets of solutions of a higher diversity than those generated by DivFF on the same problems, with the same distance metric. Perhaps the most important reason for this is the fact that the search space of DivFF is limited through heuristic-based selections specific to heuristic-search planning. While boosting efficiency, these limitations severely restrict the opportunities for finding diverse solutions by rendering the search space itself unvaried. These observations were put to use when designing DivNDP: in order to ensure its capability to produce highly diverse plans sets, preliminary filtering and other search space limitations were eliminated.

All described and evaluated planning systems successfully generate sets of demonstrably diverse solutions to planning problems, using varied criteria for assessing solution dissimilarity.

10.1   Future Work

Future work possibilities are manifold. They include extending the diverse planning framework and enhancing it with other Artificial Intelligence techniques, such as machine learning.

In this work, the diversity of generated solutions has been the primary concern. However, one would not want to encourage diversity at the detriment of solution quality, i.e. generate a set of solutions which are diverse, but of such low average quality as to make most of them unusable. An immediate difficulty in taking quality considerations into account when generating diverse solutions lies in the fact that assessing planning solution quality is in itself no easy matter. The traditional approach of considering lower-size solutions to be of higher quality than larger ones is not guaranteed to be meaningful in all domains: a longer plan is not always less preferable than a shorter one (e.g. in an aerial transportation domain, a longer plan including more security measures may be preferable due to lower risk levels). On the other hand, in dynamic domains like Wargus, it is difficult to assess the quality of the solutions based on their low-level structure before actually running them in the environment (whether a plan will succeed or not will strongly depend on exterior factors such as characteristics of the map not known at planning time).

A possible approach that could be taken is to define qualitative measures of solution quality similarly to how qualitative measures for plan comparison have

been defined herein. The composite solution-generation criterion can be easily extended to include this additional component, as in Equation 26 below (where α + β + γ = 1):

$$ExtendedEvalCrit(\pi, \Pi) =$$
$$\alpha SolAdequacy(\pi) + \beta RelDiv(\pi, \Pi) + \gamma SolQuality(\pi) \qquad (26)$$

Such an approach has been taken by Srivastava *et al*. (2007), who make use of the quality considerations specific to LPG.

The general framework could also be extended by including planning preferences (Nguyen *et al*., 2012) within it, so that the generated sets of solutions are diverse only insofar as it is possible within the specific requirements of users or various other restrictions.

Another question is whether it would be possible to produce good solution distance metrics automatically. This might be achieved through machine-learning techniques, which have been previously employed for learning measures of similarity in various fields, such as case-based reasoning (Ricci and Avesani, 1995) and search and classification tasks in social media (Becker, Naaman, and Gravano, 2010). Learning distance metrics automatically might be a way of addressing the

difficulties humans sometimes encounter in identifying meaningful differences between items (Dunbar, 2001).

This work focuses on algorithms for diverse-plan generation and on two large categories of distance metrics (quantitative and qualitative), but not on creating good qualitative distance metrics for specific application domains, nor on issues pertaining to encoding complex distance metrics efficiently. Consequently, the qualitative distance metrics presented herein are relatively simple. Future work could, therefore, focus on building complex qualitative distance metrics and perhaps testing these with implementations of the diverse planning framework built on top of highly-efficient planning systems.

I have addressed diversity in automated planning conducted exclusively by artificially intelligent systems, not within mixed-initiative systems based on human/AI collaboration. Diversity-aware mixed-initiative planning is, therefore, another future research direction: such systems could assist human planners in keeping track of varied possibilities in domains characterized by vast solution spaces. While the artificially intelligent collaborator in a mixed-initiative context may not have the freedom to generate complete diverse solutions by itself, it can still propose diverse partial solutions matching goals or subgoals set by the human collaborator.

While diversity in interactive storytelling has been touched upon briefly herein, merely as a demonstration of a possible application of diverse planning techniques,

there are plentiful possibilities for more work on diverse interactive storytelling. One challenge is that of enforcing diversity within the fixed landmarks employed by some interactive-storytelling planning systems (Porteous, Cavazza, and Charles, 2010b). This is related to diverse planning with user preferences/restrictions as mentioned above.

Finally, as there have so far been no attempts to solve nondeterministic planning problems through case-based approaches to planning, both nondeterministic case-based planning and **diverse** nondeterministic case-based planning remain open research directions.

In addition to these high-level extensions of the diverse planning framework and research directions which, while related, depart significantly from this work, the actual implemented systems presented herein would also benefit from various enhancements.

Firstly, in order to enable DivNDP to produce highly diverse solution sets, JavaFF (the classical planner DivNDP is based on) has been modified so as to only use the complete search algorithm that normally comes into play only after a more efficient, incomplete algorithm has failed. This has dramatically improved diversity in the experimental evaluation. On the domain it was tested on, it also did not cause any efficiency issues. This, however, may not be the case on other application domains. It might, therefore, be necessary to devise methods of maintaining efficiency while using DivNDP with complete search.

One possibility of achieving balance between result diversity and planning efficiency is partially inspired by the Adaptive Selection algorithm of McGinty and Smyth, 2003 (see Chapter 10.1.1). Based on how difficult it seems to find a solution, a decision could be made during the planning process whether to keep using the composite criterion or to switch to the regular FF heuristic. It would also be possible to maintain a diversity threshold so that, once the partial solution has reached the threshold, the composite criterion is abandoned, and the system switches to the regular solution-adequacy criterion. On the other hand, re-computing this diversity value at each planning stage might itself prove costly.

Efficiency has not been a primary objective in this work, and the JavaFF planner, which the implementations of DivFF and DivNDP both use, is not a particularly efficient implementation of FF, as it is intended primarily for pedagogical purposes, not for tasks requiring high efficiency. However, the presented framework for diverse heuristic planning is not planner-specific, hence the modifications it describes can be integrated within any heuristic planner. It would, therefore, be possible to test the proposed methods with state-of-the-art, highly-efficient planners, such as LAMA (Richter and Westphal, 2010).

In case-based planning, DivCBP introduces diversity at the retrieval stage. Another possibility would be to do so at the reuse stage, by modifying the adaptation algorithms, rather than the retrieval criterion, so as to take relative

diversity into account. This might be advantageous when using large case bases, which make it costly to compute relative diversity for every case-base entry.

Furthermore, while DivCBP is a transformational-analogy diverse case-based planner, it would be interesting to investigate whether it would be feasible to introduce diversity in the context of derivational-analogy case-based planning as well. This might prove challenging, as these types of planners generally attempt to re-run previous planning processes as closely as possible to the recorded trace. Diversity could perhaps be introduced when generating those portions of plans for which retrieved derivational traces cannot be used, and which have to be produced from scratch.

Within the skill panoply of artificial intelligences, this work contributes to the ability to come up with several different courses of action to solve a problem. This skill has as prerequisite the more general one of discerningly comparing between courses of action: in the AI world, comparing things is not an ability to be taken for granted, but one to be gradually conquered.

Agents equipped with diverse-planning skills could keep us (and themselves) supplied with alternative approaches to fall back on when things do not turn out as anticipated; point out diverse, potentially surprising ways in which our data may be at risk of intrusion; act as computer-game characters who, although not possessed of real personalities, can exhibit varied behavior roughly mimicking personality

variation; point out options we may have overlooked when collaborating with us on a planning task. There are many ways in which they could expand and borrow from their AI relatives' abilities (a valuable new skill would be that of learning by themselves how to differentiate between different plans, making them more autonomous and helpful in new ways). Irrespective of the techniques used to attain it, there will always be room for diversity in Artificial Intelligence.

Bibliography

[Aamodt and Plaza, 1994] Aamodt, A. and Plaza, E. 1994. Case-Based Reasoning: Foundational Issues, Methodological Variations, and System Approaches, *Artificial Intelligence Communications* 7: 1, 39-52.

[Aha, Breslow, and Muñoz-Avila, 2000] Aha, D.W.; Breslow, L.A.; and Muñoz-Avila, H. 2000. Conversational Case-Based Reasoning. *Applied Intelligence*, 14:9–32.

[Arkin *et al*., 2003] Arkin, R.C.; Fujita, M.; Takagi, T,; and Hasegawa, R. 2003. An Ethological and Emotional Basis for Human-Robot Interaction. *Robotics and Autonomous Systems* 42(3-4): 191-201.

[Au, Muñoz-Avila, and Nau, 2002] Au, T.C.; Muñoz-Avila, H.; and Nau, D.S. 2002. On the Complexity of Plan Adaptation by Derivational Analogy in a Universal Classical Planning Framework. In Proceedings of the 6th European Conference on Advances in Case-Based Reasoning (ECCBR 2002) LNCS, vol. 2416, 13-27. Springer, Berlin.

[Bartheye and Jacopin, 2009] Bartheye, O., and Jacopin, E. 2009. A Real-Time PDDL-based Planning Component for Video Games. In Proceedings of the Fifth Artificial Intelligence for Interactive Digital Entertainment Conference (AIIDE-09), 130-135. AAAI Press.

[Becker, Naaman, and Gravano, 2010] Becker, H.; Naaman, M.; and Gravano, L. 2010. Learning Similarity Metrics for Event Identification in Social Media. Proceedings of the Third International Conference on Web Search and Web Data Mining (WSDM 2010), 291-300. ACM.

[Berger, Guilford, and Christensen, 1957] Berger, R. M.; Guilford, J. P.; and Christensen, P. R. 1957. A Factor-Analytic Study of Planning Abilities. *Psychological Monographs*, 71, 1–31.

[Berger and Bell, 1988] Berger, C. R., and Bell, R. A. 1988. Plans and the Initiation of Social Relationships. *Communication Research*, 15, 217 – 235.

[Berger and DiBattista, 1993] Berger, C.R., and DiBattista, P. 1993. Communication Failure and Plan Adaptation: If at First You Don't Ducceed, Say It Louder and Slower. *Communication Monographs*, 60(3), 220-238.

[Berger, 2007] Berger, C. R. 2007. Plans, Planning, and Communication Effectiveness. In: Whaley, B.B., Samter, W. (Eds.), Explaining Communication: Contemporary Theories and Exemplars, 149-163. Lawrence Erlbaum Associates Publishers, Mahwah, NJ.

[Bettina, 1999] Bettina, H. 1999. Plan Modification: An Examination of Cognitive Planning Theory. *Communication & Cognition*, 32(3-4), 201-221.

[Blum and Furst, 1997] Blum, A., and Furst, M. 1997. Fast Planning Through Planning Graph Analysis. *Artificial Intelligence* 90:281-300.

[Boddy *et al.*, 2005] Boddy, M.; Gohde, J.; Haigh, T.; and Harp, S. 2005. Course of Action Generation for Cyber Security using Classical Planning. In Proceedings of the Fifteenth International Conference on Automated Planning and Scheduling (ICAPS 2005). AAAI Press.

[Bresina *et al*., 2005] Bresina, J.L.; Jónsson, A.K.; Morris, P.H.; and Rajan, K. 2005. Mixed-Initiative Activity Planning for Mars Rovers. In Proceedings of the Nineteenth International Joint Conference on Artificial Intelligence (IJCAI 2005), 1709-1710. AAAI Press/IJCAI.

[Bridge and Ferguson, 2002] Bridge, D.G., and Ferguson, A. 2002. Diverse Product Recommendations Using an Expressive Language for Case Retrieval. In Proceedings of the Sixth European Conference on Case-Based Reasoning (ECCBR 2002), LNCS, vol. 2416, 43-57. Springer.

[Bridge and Kelly, 2006] Bridge, D., and Kelly, J.P. 2006. Ways of Computing Diverse Collaborative Recommendations. In Proceedings of the Fourth International Conference on Adaptive Hypermedia and Adaptive Web-Based Systems. Springer.

[Bryce *et al*., 2007] Bryce, D.; Cushing, W.; and Kambhampati, S. 2007. Model-lite Planning: Diverse Multi-option Plans and Dynamic Objective Functions. ICAPS 2007 Workshop on Planning and Plan Execution for Real World Systems.

[Cavazza, Charles, and Mead, 2002] Cavazza, M.; Charles, F; and Mead, S.J. 2002. Sex, Lies, and Video Games: an Interactive Storytelling Prototype. *AAAI Technical Report SS-02-01*, 13-17. AAAI Press.

[Cavazza and Charles, 2005] Cavazza, M., and Charles, F. 2005. Dialogue Generation in Character-based Interactive Storytelling. In Proceedings of the First Artificial Intelligence and Interactive Digital Entertainment Conference (AIIDE 2005), 21-16. AAAI Press.

[Cavazza *et al*., 2009] Cavazza, M.; Pizzi, D.; Charles, F.; Vogt, T.; and André, E. 2009. Emotional Input for Character-based Interactive Storytelling. In Proceedings of the Eighth International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS 2009), 313-320. IFAAMAS.

[Chang *et al*., 2011] Chang, Y-H.; Maheswaran, R.T.; Levinboim, T.; and Rajan, V. 2011. Learning and Evaluating Human-Like NPC Behaviors in Dynamic Games. In Proceedings of the Seventh AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment (AIIDE 2011), 8-13. AAAI Press.

[Cimatti *et al.*, 2003] Cimatti, A.; Pistore, M.; Roveri, M; and Traverso, P. 2003. Weak, Strong, and Strong Cyclic Planning via Symbolic Model Checking. *Artificial Intelligence*, 147(1-2):35–84.

[Coles *et al.*, 2008] Coles, A. I.; Fox, M.; Long, D.; and Smith, A. J. 2008. Teaching Forward-Chaining Planning with JavaFF. Colloquium on AI Education, Twenty-Third AAAI Conference on Artificial Intelligence.

[Cox, Muñoz-Avila, and Bergmann, 2005] Cox, M.T.; Muñoz-Avila, H; and Bergmann, R. 2005 Case-Based Planning. *The Knowledge Engineering Review,* 20.

[Coman and Muñoz-Avila, 2010] Coman, A., and Muñoz-Avila, H. 2010. Case-based Plan Diversity. In Proceedings of the Eighteenth International Conference on Case-Based Reasoning (ICCBR 2010), LNCS, vol. 6176, 66-80. Springer.

[Coman and Muñoz-Avila, 2011a] Coman, A., and Muñoz-Avila, H. 2011a. Generating Diverse Plans Using Quantitative and Qualitative Plan Distance Metrics. In Proceedings of the Twenty-Fifth AAAI Conference on Artificial Intelligence (AAAI 2011), 946-951. AAAI Press.

[Coman and Muñoz-Avila, 2011b] Coman, A., and Muñoz-Avila, H. 2011b. Qualitative vs. Quantitative Plan Diversity in Case-Based Planning. In Proceedings of the Nineteenth International Conference on Case-Based Reasoning (ICCBR 2011), LNCS, vol. 6880, 32-46. Springer.

[Coman and Muñoz-Avila, 2012a] Coman, A., and Muñoz-Avila, H. 2012a. Diverse Plan Generation by Plan Adaptation and by First-Principles Planning: A Comparative Study. In Proceedings of the Twentieth International Conference on Case-Based Reasoning (ICCBR 2012), LNCS, vol. 7466, 32-46. Springer.

[Coman and Muñoz-Avila, 2012b] Coman, A., and Muñoz-Avila, H. 2012b. Plan-Based Character Diversity. In Proceedings of the Eighth AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment (AIIDE 2012), 118-123. AAAI Press.

[Dasgupta et al., 1999] Dasgupta, P.; Chakrabarti, P.P.; and DeSakar, S.C. 1999. Multiobjective Heuristic Search: An Introduction to Intelligent Search Methods for Multicriteria Optimization. Vieweg, Braunschweig/Wiesbaden.

[DeJong, 1975] DeJong, K.A. 1975. Analysis of the Behavior of a Class of Genetic Adaptive Systems. Ph.D. thesis, University of Michigan, Ann Arbor.

[Díaz-Agudo et al., 2008] Díaz-Agudo, B.; Plaza, E.; Recio-García, J.A.; Arcos, J.L. 2008. Noticeably New: Case Reuse in Originality-Driven Tasks. In Proceedings of the Ninth European Conference on Advances in Case-Based Reasoning (ECCBR 2008), LNCS, vol. 5239, 165-179. Springer.

[Dillard, 2008] Dillard, J. P. 2008. Goals–Plans–Action Theory of Message Production. In: Baxter, L. A., Braithwaite, D. O. (Eds.) Engaging Theories in Interpersonal Communication: Multiple Perspectives, SAGE Publications, Inc.

[Dunbar, 2001] Dunbar, K. 2001. The Analogical Paradox: Why Analogy Is So Easy in Naturalistic Settings, Yet So Difficult in the Psychology Laboratory. In: Gentner, D.; Holyoak, K. J.; and Kokinov, B. (Eds.) Analogy: Perspectives from Cognitive Science. Cambridge, MA, MIT Press.

[Eiter *et al*., 2011] Eiter, T.; Erdem, E.; Erdogan, H; and Fink, M. 2012. Finding Similar/Diverse Solutions in Answer Set Programming. arXiv:1108.3260. *To appear* in *Theory and Practice of Logic Programming*.

[Fairclough, 2004] Fairclough, C. 2004. Story Games and the OPIATE System: Using Case-Based Planning for Structuring Plots with an Expert Story Director Agent and Enacting Them in a Socially Simulated Game World. Ph.D. dissertation, University of Dublin, Trinity College.

[Felner *et al*., 2007] Felner, A.; Stern, R.; Rosenschein, J.S.; Pomeransky, A. 2007. Searching for Close Alternative Plans. *Autonomous Agents and Multi-Agent Systems* 14(3): 209.

[Forbus, Gentner, and Law, 1995] Forbus, K.D.; Gentner, D.; and Law, K. 1995. MAC/FAC: A Model of Similarity-Based Retrieval. *Cognitive Science* 19(2): 141-205.

[Fox *et al*., 2006] Fox, M.; Gerevini, A.; Long, D.; and Serina, I. 2006. Plan Stability: Replanning versus Plan Repair. In Proceedings of the Sixteenth International Conference on Automated Planning and Scheduling (ICAPS 2006), 212–221. AAAI Press.

[Fu *et al*., 2011] Fu, J.; Ng, V.; Bastani, F.B.; and Yen, I. 2011. Simple and Fast Strong Cyclic Planning for Fully-Observable Nondeterministic Planning Problems. In Proceedings of the Twenty-second International Joint Conference on Artificial Intelligence (IJCAI 2011), 1949-1954. AAAI Press/IJCAI.

[Gelfond and Lifschitz, 1991] Gelfond, M., and Lifschitz, V. 1991. Classical Negation in Logic Programs and Disjunctive Databases. *New Generation Computing* 9(3/4): 365-386.

[Gerevini and Serina, 2000] Gerevini, A., and Serina, I. 2000. Fast Plan Adaptation through Planning Graphs: Local and Systematic Search Techniques. In Proceedings of the Fifth International Conference on Artificial Intelligence Planning Systems (AIPS 2000), 112-121. AAAI Press.

[Gerevini, Saetti, and Serina, 2003] Gerevini, A.; Saetti, A.; and Serina, I. 2003. Planning Through Stochastic Local Search and Temporal Action Graphs. Journal of Artificial Intelligence Research 20: 239-290.

[Gerevini and Serina, 2010] Gerevini, A., and Serina, I. 2010. Efficient Plan Adaptation through Replanning Windows and Heuristic Goals. *Fundamenta Informaticae* 2(3-4):287-323.

[Ghallab, Nau, and Traverso, 2004] Ghallab, M.; Nau, D.S.; and Traverso, P. 2004. Automated Planning: Theory and Practice. Morgan Kaufmann.

[Guilherme da Silva, Ciarlini, and Siqueira, 2010] Guilherme da Silva, F.A.; Ciarlini, A.E.M.; Siqueira, S.W.M. 2010. Nondeterministic Planning for Generating Interactive Plots. In Proceedings of the Twelfth Ibero-American Conference on AI (IBERAMIA 2010). LNCS vol. 6433, 133–143. Springer, Heidelberg.

[Hammond, 1990] Hammond, K. J. 1990. Case-based Planning: A Framework for Planning from Experience. *Cognitive Science* 14(3): 385–443.

[Hansen and Zilberstein, 2001] Hansen, E., and Zilberstein, S. 2001. LAO*: A Heuristic-Search Algorithm That Finds Solutions with Loops. *Artificial Intelligence* 129(1–2):35–62.

[Hayes-Roth and Hayes-Roth, 1979] Hayes-Roth, B., and Hayes-Roth, F. 1979. A Cognitive Model of Planning. *Cognitive Science*, 3(4):275–310.

[Hebrard et *al*., 2005] Hebrard, E.; Hnich, B.; O'Sullivan, B.; and Walsh, T. 2005. Finding Diverse and Similar Solutions in Constraint Programming. In Proceedings of the Twentieth National Conference on Artificial Intelligence (AAAI 2005), 372-377. AAAI Press.

[Hoffmann and Nebel, 2001] Hoffmann, J., and Nebel, B. 2001. The FF Planning System: Fast Plan Generation Through Heuristic Search, *Journal of Artificial Intelligence Research*, 14:253-302.

[Hogg, Kuter, and Muñoz-Avila, 2009] Hogg, C.; Kuter, U.; and Muñoz-Avila, H. 2009. Learning Hierarchical Task Networks for Nondeterministic Planning Domains. In Proceedings of the Twenty-first International Joint Conference on Artificial Intelligence (IJCAI 2009), 1708-1714. AAAI Press/IJCAI.

[Houlette and Fu, 2003] Houlette, R., and Fu, D. 2003. The Ultimate Guide to FSMs in Games. In: Rabin, S. (Ed.) *AI Game Programming Wisdom 2*. Charles River Media.

[Kambhampati and Srivastava, 1995] Kambhampati, S., and Srivastava B. 1995. Universal Classical Planner: An Algorithm for Unifying State-Space and Plan-

Space Planning. Proceedings of the European Workshop on Planning (EWSP 1995).

[Keane, 1996] Keane, M. T. 1996. On Adaptation in Analogy: Tests of Pragmatic Importance and Adaptability in Analogical Problem Solving. *Quarterly Journal of Experimental Psychology*, 49:1062–1085.

[Kissmann and Edelkamp, 2009] Kissmann, P., and Edelkamp, S. 2009. Solving Fully-Observable Nondeterministic Planning Problems via Translation into a General Game. In Proceedings of KI, LNCS, vol. 5803, 1–8, Springer.

[Klahr, Fay, and Dunbar, 1993] Klahr, D.; Fay, A. L.; and Dunbar, K. 1993. Heuristics for Scientific Experimentation: A Developmental Study. *Cognitive Psychology*, 25:111–146.

[Ko, 2002] Ko, S. 2002. An Empirical Analysis of Children's Thinking and Learning in a Computer Game Context. *Educational Psychology*, 22(2):219-233.

[Koedinger and Anderson, 1990] Koedinger, K.R., and Anderson, J.R. 1990. Abstract Planning and Perceptual Chunks: Elements of Expertise in Geometry. *Cognitive Science*, 14, 511-550.

[Kuchibatla and Muñoz-Avila, 2006] Kuchibatla, V., and Muñoz-Avila, H. 2006. An Analysis on Transformational Analogy: General Framework and Complexity.

In Proceedings of the Eighth European Conference on Advances in Case-Based Reasoning (ECCBR 2006), LNCS, vol. 4106, 458-473. Springer, Heidelberg.

[Kuter *et al*., 2008] Kuter, U.; Nau, D. S.; Reisner, E.; and Goldman, R. P. 2008. Using Classical Planners to Solve Nondeterministic Planning Problems. In Proceedings of the Eighteenth International Conference on Automated Planning and Scheduling (ICAPS 2008). AAAI Press.

[Lin and Walker, 2011] Lin, G.I., and Walker, M.A. 2011. All the World's a Stage: Learning Character Models from Film. In Proceedings of the Seventh AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment, 46-52. AAAI Press.

[Lussier *et al*., 2007] Lussier, B.; Gallien, M.; Guiochet, J.; Ingrand, F.; Killijian, M.O.; and Powell, D. 2007. Planning with Diversified Models for Fault-Tolerant Robots. In Proceedings of the Seventeenth International Conference on Automated Planning and Scheduling (ICAPS 2007), 216-223. AAAI Press.

[Mandow and Pérez de la Cruz, 2003] Mandow, L., and Pérez de la Cruz, J.L. 2003. Multicriteria Heuristic Search. *European Journal of Operational Research*, 150:253–280, 2003.

[Mattmüller *et al*., 2010] Mattmüller, R.; Ortlieb, M.; Helmert, M.; and Bercher, P. 2010. Pattern Database Heuristics for Fully Observable Nondeterministic Planning.

In Proceedings of the Twentieth International Conference on Automated Planning and Scheduling (ICAPS 2010), 105–112. AAAI Press.

[Mauldin, 1984] Mauldin, M.L. 1984. Maintaining Diversity in Genetic Search. In Proceedings of the National Conference on Artificial Intelligence (AAAI 1984), 247-250. AAAI Press.

[McCoy *et al*., 2010] McCoy, J.; Treanor, M.; Samuel, B.; Tearse, B.; Mateas, M.; and Wardrip-Fruin, N. 2010. The Prom: An Example of Socially-Oriented Gameplay. In Proceedings of the Sixth AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment (AIIDE 2010), 221-222. AAAI Press.

[McGinty and Smyth, 2003] McGinty, L., and Smyth, B. 2003. On the Role of Diversity in Conversational Recommender Systems. In Proceedings of the Fifth International Conference on Case-Based Reasoning (ICCBR 2003), LNCS, vol. 2689, 276–290. Springer.

[McSherry, 2002] McSherry, D. 2002. Diversity-Conscious Retrieval. In Proceedings of the Sixth European Conference on Case-Based Reasoning (ECCBR 2002), LNCS, vol. 2416, 219–233. Springer.

[Miller Henningsen *et al*., 2011] Miller Henningsen, M. L; Valde, K. S.; Russell, G. A.; and Russell, G. R. 2011. Student–Faculty Interactions About Disappointing

Grades: Application of the Goals–Plans–Actions Model and the Theory of Planned Behavior . *Communication Education*, 60(2):174 – 190.

[Molineaux, Klenk, and Aha, 2010] Molineaux, M.; Klenk, M.; and Aha, D. 2010. Goal-Driven Autonomy in a Navy Strategy Simulation. In Proceedings of the Twenty-Fourth AAAI Conference on Artificial Intelligence (AAAI 2010), 1115-1120, AAAI Press.

[Mumford, Schultz, and Van Doorn, 2001] Mumford, M. D.; Schultz, R. A.; and Van Doorn, J. R. 2001. Performance in Planning: Processes, Requirements, and Errors. *Review of General Psychology, 5*(3):213-240.

[Muñoz-Avila and Cox, 2008] Muñoz-Avila, H., and Cox, M.T. 2008. Case-Based Plan Adaptation: An Analysis and Review. *IEEE Intelligent Systems* 23(4): 75-81.

[Myers and Lee, 1999] Myers, K. L., and Lee, T. J. 1999. Generating Qualitatively Different Plans through Metatheoretic Biases. In Proceedings of the Sixteenth National Conference on Artificial Intelligence (AAAI 1999), 570–576. AAAI Press/MIT Press.

[Myers *et al*., 2002] Myers, K. L.; Tyson, W. M.; Wolverton, M. J.; Jarvis, P. A.; Lee, T. J.; and desJardins, M. 2002. PASSAT: A User-centric Planning Framework. In Proceedings of the 3rd Intl. NASA Workshop on Planning and Scheduling for Space.

[Myers, 2006] Myers, K. L. 2006. Metatheoretic Plan Summarization and Comparison. In Proceedings of the Sixteenth International Conference on Automated Planning and Scheduling (ICAPS 2006), 182-192. AAAI Press.

[Nebel and Koehler, 1995] Nebel, N., and Koehler, J. 1995. Plan Reuse versus Plan Generation: A Theoretical and Empirical Analysis, *Artificial Intelligence*, 76:427–454.

[Newell, Shaw, and Simon, 1959] Newell, A.; Shaw, J.C.; and Simon, H.A. 1959. Report on a General Problem-Solving Program. In Proceedings of the International Conference on Information Processing, 256–264.

[Newell and Simon, 1972] Newell, A., and Simon, H.A. 1972. Human Problem Solving. Prentice-Hall, Oxford, England.

[Newell, 1978] Newell, A. 1978. Harpy, Production Systems and Human Cognition. *Technical Report*, Carnegie Mellon University.

[Nguyen *et al*., 2012] Nguyen, T.; Do, M.; Gerevini, A.; Serina, I.; Srivastava, B.; and Kambhampati, S. 2012. Generating Diverse Plans to Handle Unknown and Partially Known User Preferences. *Artificial Intelligence* 190:1-31.

[Ontañón *et al*., 2010] Ontañón, S.; Mishra, K.; Sugandh, N.; and Ram, A. 2010. On-line Case-Based Planning. *Computational Intelligence Journal* 26(1): 84-119.

[Orkin, 2003] Orkin, J. 2003. Applying Goal-Oriented Action Planning to Games. In: Rabin, S. (Ed.) *AI Game Programming Wisdom 2*. Charles River Media.

[Paul *et al*., 2010] Paul, R.; Charles, D.; McNeill, M.; and McSherry, D. 2010. MIST: An Interactive Storytelling System with Variable Character Behavior. In Proceedings of the Third Joint Conference on Interactive Digital Storytelling (ICIDS 2010), LNCS vol. 6432, 4-15. Springer.

[Porteous, Cavazza, and Charles, 2010a] Porteous, J.; Cavazza, M.; and Charles, F. 2010a. Narrative Generation through Characters' Point of View. In Proceedings of the Ninth International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2010), 1297-1304. IFAAMAS.

[Porteous, Cavazza, and Charles, 2010b] Porteous, J.; Cavazza, M.; and Charles, F. 2010. Applying Planning to Interactive Storytelling: Narrative Control using State Constraints. *ACM Transactions on Intelligent Systems and Technology* 1(2):1-21.

[Ricci and Avesani, 1995] Ricci, F., and Avesani, P. 1995. Learning a Local Similarity Metric for Case-based Reasoning. In Proceedings of the First International Conference on Case-Based Reasoning (ICCBR 1995), LNCS, vol. 1010, Springer.

[Richter and Westphal, 2010] Richter, S., and Westphal, M. 2010. The LAMA Planner: Guiding Cost-Based Anytime Planning with Landmarks. *Journal of Artificial Intelligence Research* 39:127–177.

[Roberts *et al*., 2012] Roberts, M.; Howe, A.E.; Ray, I.; and Urbanska, M. 2012. Using Planning for a Personalized Security Agent. AAAI 2012 Workshop on Problem Solving Using Classical Planners.

[Rossi, Van Beek, and Walsh, 2006] Rossi, F.; Van Beek, P.; and Walsh, T. 2006. Handbook of Constraint Programming. Elsevier.

[Russell and Norvig, 2009] Russell, S., and Norvig, P. 2009. Artificial Intelligence: A Modern Approach, 3rd Edition. Prentice Hall.

[Shell, Hernandez Rubio, and Quiroga Barro, 1994] Shell, P.; Hernandez Rubio, J.A.; Quiroga Barro, G. 1994. Improving Search through Diversity. In Proceedings of the Twelfth National Conference on Artificial Intelligence (AAAI 1994), 1323-1328. AAAI Press/MIT Press.

[Shimazu, 2001] Shimazu, H. 2001. ExpertClerk: Navigating Shoppers' Buying Process with the Combination of Asking and Proposing. In Proceedings of the Seventeenth International Joint Conference on Artificial Intelligence (IJCAI-2001), 1443–1448. Morgan Kaufmann, Seattle, Washington, USA.

[Smyth and Keane, 1998] Smyth, B., and Keane, M.T. 1998. Adaptation-Guided Retrieval: Questioning the Similarity Assumption in Reasoning. *Artificial Intelligence* 102:249–293.

[Smyth and McClave, 2001] Smyth B., and McClave, P. 2001. Similarity vs. Diversity. In Proceedings of the Fourth International Conference on Case-Based Reasoning (ICCBR 2001), LNCS, vol. 2080, 347–361. Springer.

[Spalazzi, 2001] Spalazzi, L. 2001. A Survey on Case-Based Planning. *Artificial Intelligence Review* 16(1):3–36. Springer Netherlands.

[Srivastava *et al*., 2007] Srivastava, B.; Kambhampati, S; Nguyen, T.; Do, M.; Gerevini, A.; and Serina, I. 2007. Domain Independent Approaches for Finding Diverse Plans. In Proceedings of the 20th International Joint Conference on Artificial Intelligence (IJCAI 2007), 2016-2022. AAAI Press/IJCAI.

[Sroka and Long, 2012] Sroka, M., Long. D. 2012. Exploring Metric Sensitivity of Planners for Generation of Pareto Frontiers. Proceedings of the Sixth Starting AI Researchers' Symposium (STAIRS 2012), 306-317. IOS Press.

[Stewart and White, 1991] Stewart, B.S., and White, C.C. 1991. Multiobjective A*. *Journal of the ACM*, 38(4):775–814.

[Strong *et al*., 2007] Strong, C.R.; Mehta, M.; Mishra, K.; Jones, A.; and Ram, A. 2007. Emotionally Driven Natural Language Generation for Personality Rich

Characters in Interactive Games. In Proceedings of the Third Artificial Intelligence and Interactive Digital Entertainment Conference (AIIDE 2007), 98-100. AAAI Press.

[Szita, Ponsen, and Spronck, 2009] Szita, I.; Ponsen, M.; and Spronck, P. 2009. Effective and Diverse Adaptive Game AI. *IEEE Transactions on Computational Intelligence and AI in Games* 1(1):16-27.

[Tate, Dalton, and Levine, 1998] Tate, A.; Dalton, J.; and Levine, J. 1998. Generation of Multiple Qualitatively Different Plan Options. In Proceedings of the Fourth International Conference on Artificial Intelligence Planning Systems (AIPS 1998), 27-35. AAAI Press.

[Thue *et al*., 2010] Thue, D.; Bulitko, V.; Spetch, M.; and Webb, M. 2010. Socially Consistent Characters in Player-Specific Stories. In Proceedings of the Sixth AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment (AIIDE 2010), 198-203. AAAI Press.

[van der Krogt and de Weerdt, 2005] van der Krogt, R., and de Weerdt, M. 2005. Plan Repair as an Extension of Planning. In Proceedings of the Fifteenth International Conference on Automated Planning and Scheduling (ICAPS 2005), 161-170. AAAI Press.

[Veloso, 1994] Veloso, M. 1994. Planning and Learning by Analogical Reasoning. Springer Verlag, Berlin.

[Waldron, 1990] Waldron, V.R. 1990. Constrained Rationality: Situational Influences on Information Acquisition Plans and Tactics. *Communication Monographs*, 57(3):184-201.

[Zhang, Coenen, and Leng, 2002] Zhang, L.; Coenen, F.; Leng, P.H. 2002. An Experimental Study of Increasing Diversity for Case-Based Diagnosis. In Proceedings of the Sixth European Conference on Advances in Case-Based Reasoning (ECCBR 2002), LNCS vol. 2416, 448-459. Springer.

Vita

I was born on November 9th, 1984, in Bucharest, Romania. My parents are Liana and Liviu Coman. I earned my Bachelor of Science Degree from the Romanian-American University in Bucharest, Romania, in June 2007. In January 2011, I earned my Master of Science Degree from Lehigh University in Bethlehem, Pennsylvania. As a graduate student at Lehigh University, I worked as Teaching Assistant and Research Assistant in the InSyTe lab, coordinated by Professor Héctor Muñoz-Avila.

Publications

Coman, A., and Muñoz-Avila, H. 2012. Plan-Based Character Diversity. In Proceedings of the Eighth AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment (AIIDE 2012), 118-123. AAAI Press.

Coman, A., and Muñoz-Avila, H. 2012. Diverse Plan Generation by Plan Adaptation and by First-Principles Planning: A Comparative Study. In Proceedings of the Twentieth International Conference on Case-Based Reasoning (ICCBR 2012), LNCS, vol. 7466, 32-46. Springer.

Coman, A., and Muñoz-Avila, H. 2012. Creating Diverse Storylines by Reusing Plan-Based Stories. Proceedings of the ICCBR 2012 Workshops (TRUE and Story Cases: Traces for Reusing Users' Experiences - Cases, Episodes, and Stories Workshop), 213-221.

Coman, A. 2012. Solution Diversity in Case-Based and Generative Planning: Research Summary. Proceedings of the ICCBR 2012 Workshops (Doctoral Consortium), 237-239.

Coman, A. 2012. Solution Diversity in Planning: Extended Abstract. Proceedings of the Seventeenth AAAI/SIGART (AAAI 2012) Doctoral Consortium, 2386-2387.

Coman, A., and Muñoz-Avila, H. 2011. Qualitative vs. Quantitative Plan Diversity in Case-Based Planning. In Proceedings of the Nineteenth International Conference on Case-Based Reasoning (ICCBR 2011), LNCS, vol. 6880, 32-46. Springer.

Coman, A., and Muñoz-Avila, H. 2011. Generating Diverse Plans Using Quantitative and Qualitative Plan Distance Metrics. In Proceedings of the Twenty-Fifth AAAI Conference on Artificial Intelligence (AAAI 2011), 946-951. AAAI Press.

Coman, A., and Muñoz-Avila, H. 2010. Case-based Plan Diversity. In Proceedings of the Eighteenth International Conference on Case-Based Reasoning (ICCBR 2010), LNCS, vol. 6176, 66-80. Springer.