Theoretical Computer Science 38 (1985) 193-222
North-Holland

AN INVESTIGATION OF CONTROLS FOR CONCURRENT SYSTEMS BASED ON ABSTRACT CONTROL LANGUAGES

H.D. BURKHARD

Sektion Mathematik, Humboldt-Universität, 1086 Berlin, Postfach 1297, German Democratic Republic

Communicated by G. Mirkowska
Received June 1984
Revised November 1984

Abstract. The behaviour of the controlled system determines the control. This concise statement summarizes our approach to the investigation of controls. Using abstract languages to define the behaviour and subbehaviour of a system, and therewith the behaviour of the uncontrolled and of the controlled system, we are able to describe and to study different types of control rules and properties to be realized by control like deadlock avoidance, liveness and fairness.

Introduction

Control is one of the central problems in studies of concurrent systems and programs. It is inherent in conflict resolution, scheduling, synchronization, program semantics, wherever decisions concerning choices are made. There is a common understanding about control, but general definitions and considerations are missing.

From its use in literature, two aspects appear: application of certain control rules (queues, priorities, choice of maximal sets of concurrently performable actions, etc), and controls which are defined as restrictions of the behaviour in order to enforce properties like deadlock avoidance, termination and fairness. The second approach may be misleading, for example: 'The choice of all fair executions' appears as an obscure notation from the viewpoint of control (cf. Section 6 of this paper). Both aspects should be considered on a common base since realizations of properties by control rules (e.g. fairness by queues) are an important subject. Moreover, some notions can appear under both aspects: we may have controls realized by priorities and controls realizing properties.

Two observations are essential in order to come to general considerations of controls:

(1) Control is considered as a restriction with respect to the behaviour of the system to be controlled.

(2) Each restriction of the behaviour of the uncontrolled system determines a control in the sense that all decisions of the control are well-defined.

This correspondence between behaviour and control is employed for our purposes. Now it depends on the descriptions of the behaviour which problems of control can be examined. As we shall show in this paper, abstract languages describing the external behaviour are a convenient tool for many such problems. Since control may influence the behaviour at any time, the languages are supposed to be closed with respect to initial segmentation.

Special systems can be examined by the corresponding families of languages using the notion of control principles (Sections 1-3). Special emphasis is put on the regular languages and on the languages of firing sequences of finite Petri nets, as well as on the problems of deadlock avoidance, liveness and fairness. While many considerations can be performed using only the languages without reference to the structure of a system, the languages are supposed to be given in a suitable form with respect to the decidability results, etc.

Thus having a suitable framework to speak about controls, we can compare controls of different types (Section 3). We are able to investigate stepwise refinements of controls in order to realize different properties. We shall show that the order of such refinements plays an important role (Section 4). In general one has to take into account that properties are not preserved under controls, and verification results for the uncontrolled system need not be relevant for the controlled system.

Section 5 investigates the problem whether all executions corresponding to a special property can be realized by a uniform control. Such controls do, in general, not exist for fairness properties, but they do exist and are characterized for deadlock avoidance and liveness.

The last two sections deal with decidability results (existence of controls, realizations of controls by finite automata as control devices, properties of automata controlled systems). It happens that the generation of controls may be preferred to the analysis of controlled systems.

The following notations are used:

T^* (T^ω) is the set of all finite (infinite) sequences over the alphabet T, e denotes the empty word. A sequence $u \in T^*$ is a *prefix* of $v \in T^* \cup T^\omega$ ($u \sqsubseteq v$), if there exists a sequence v' with $v = uv'$. The *closure of a language* $L \subseteq T^*$ with respect to initial segmentation (prefixes) is denoted by $\tilde{L} := \{u \mid \exists v \in L: u \sqsubseteq v\}$. The *adherence* of a language L is defined by

$$\mathrm{Adh}(L) := \{w \mid w \in T^\omega \wedge \tilde{w} \subseteq L\}, \quad \text{where } \tilde{w} := \{u \mid u \in T^* \wedge u \sqsubseteq w\}.$$

The powerset of a set A is denoted by $P(A)$; $\exists^\infty$ denotes 'infinitely many', $\forall^\infty$ denotes 'almost all'.

By $\pi_v \in (\mathbb{N} \cup \{\omega\})^T$ we denote the *Parikh-vector* of a sequence $v \in T^* \cup T^\omega$, i.e. $\pi_v(t)$ denotes the number of occurrences of t in v. By $\boldsymbol{a}$ we denote the vector $(a, \ldots, a)$ (of given dimension; $a \in \mathbb{N} \cup \{\omega\}$). Operations and relations over vectors are understood componentwise. A *transition system* S is defined by $S = (T, Q, f, q_0)$, where Q is the set of states with the initial state $q_0 \in Q$, T is the set of transition names and $f: Q \times T \to P(Q)$ is the (nondeterministic) transition function. As usually,

we put

$$f(q, e) := \{q\}, \qquad f(q, ut) := \bigcup_{q' \in f(q,u)} f(q', t),$$

for $q \in Q$, $u \in T^*$, $t \in T$.

The language L_S of S is given by $L_S := \{u \mid f(q_0, u) \neq \emptyset\}$. S is called *finite* if Q and T are finite sets. The family of languages of finite transition systems is the family of the prefix closed regular languages which we denote by PREG.

Throughout the paper we assume without loss of generality that the transition systems are initially connected, i.e. we have

$$Q = \bigcup_{u \in T^*} f(q_0, u).$$

A *Petri net* N is defined by $N = (P, T, F, V, m_0)$ where P and T are the finite nonempty sets of places and transitions, respectively. $F \subseteq (P \times T) \cup (T \times P)$ is the flow relation and $V: F \to \mathbb{N} \setminus \{0\}$ is the multiplicity function, $m_0 \in \mathbb{N}^P$ is the initial marking.

The vectors t^-, $t^+ \in \mathbb{N}^P$ are defined by

$$t^-(p) := \textbf{if } (p, t) \in F \textbf{ then } V((p, t)) \textbf{ else } 0,$$

$$t^+(p) := \textbf{if } (t, p) \in F \textbf{ then } V((t, p)) \textbf{ else } 0.$$

A transition $t \in T$ is *firable at a marking* $m \in \mathbb{N}^P$ if $t^- \leq m$, the firing of t leads to the new marking $m + \Delta t$ with $\Delta t := t^+ - t^-$. A transition sequence $u = t_1 \ldots t_n$ is firable at m if each transition t_i $(i = 1, \ldots, n)$ is firable at $m + \Delta t_1 + \cdots + \Delta t_{i-1}$. The firing of u leads from m to $m + \Delta u$ where $\Delta u := \Delta t_1 + \cdots + \Delta t_n$.

The language L_N of a Petri net N is the language of all 'firing sequences' of N, i.e. of all sequences $u \in T^*$ which are firable at m_0. The family of all those languages is denoted by FNL.

1. Control principles

We consider (controlled or uncontrolled) systems by means of their behaviour, given by languages L over a finite fixed alphabet T with $\text{card}(T) \geq 2$. We suppose these languages to be nonempty and closed with respect to initial segmentation. The control of a system is regarded as a restriction of its possibilities, thus the language L' of a controlled system is always a subset of the language L of the original (uncontrolled) system.

Definition 1.1. $\text{CONT} := \{L \mid \emptyset \neq L = \hat{L} \subseteq T^*\}$ is the family of all control languages over T.

$\text{cont}(L) := P(L) \cap \text{CONT}$ is the family of all control languages for a (control) language $L \in \text{CONT}$.

Since the behaviour of a control (the decision to be made with respect to L) is defined by a language $L' \in \text{cont}(L)$, the family $\text{cont}(L)$ describes all the possible controls of the system with the behaviour given by L. Having a special way to perform controls (like scheduling disciplines), we obtain a special subset of $\text{cont}(L)$. Having also in mind special conditions to be satisfied (like fairness, conflict resolution, etc.), we are going to study subsets of $\text{cont}(L)$.

Definition 1.2. A *control principle* is a mapping

$$c: \text{CONT} \to P(\text{CONT})$$

with $c(L) \subseteq \text{cont}(L)$ for all $L \in \text{CONT}$.

A control principle c is called *extensional* if there exists a subset $C \subseteq \text{CONT}$ such that the following holds:

$$c(L) = P(L) \cap C = \text{cont}(L) \cap C.$$

Notation. $c(C') = \bigcup_{L \in C'} c(L)$ for $C' \subseteq \text{CONT}$.

Properties like liveness can be defined by abstract languages, and a controlled system is live (or not live) independent of the original uncontrolled system. Hence, live controls can be studied by a corresponding extensional control principle (see Definition 1.4 below). Other controls like controls in order to resolve conflicts are defined by non-extensional control princples (see Definition 3.5).

Corollary 1.3. (1) *If c is extensional, then it is monotonous, i.e.*

$$L' \subseteq L'' \text{ implies } c(L') \subseteq c(L'').$$

(2) *If c is extensional, then the corresponding set C is uniquely determined by*

$$C = c(T^*) = c(\text{CONT}).$$

Definition 1.4. The extensional control principles

$$c = \text{dfr, live, imp, fair, just, pfin, preg, prec, pren, fnl}$$

are defined by the corresponding sets $C \subseteq \text{CONT}$, which for $L \in \text{CONT}$ fulfil:

(1) $L \in \text{DFR}$ iff $\forall u \in L\ \exists t \in T: ut \in L$.

(2) $L \in \text{LIVE}$ iff $\forall t \in T\ \forall u \in L\ \exists u' \in T^*: uu't \in L$.

(3) $L \in \text{IMP}$ iff $\forall w \in \text{Adh}(L): \pi_w = \omega$.

(4) $L \in \text{FAIR}$ iff $\forall t \in T\ \forall w \in \text{Adh}(L)$:
$(\exists^\infty u \sqsubseteq w: ut \in L) \to \pi_w(t) = \omega$.

(5) $L \in \text{JUST}$ iff $\forall t \in T\ \forall w \in \text{Adh}(L)$:
$(\forall^\infty u \sqsubseteq w: ut \in L) \to \pi_w(t) = \omega$.

(6)-(10) PFIN (PREG, PREC, PREN) denotes the family of all nonempty prefix-closed finite (regular, recursive, recursively enumerable) languages, FNL is the family of the firing languages of finite Petri nets.

Remark. Note that LIVE and IMP depend on the alphabet T.

By dfr(L) and live(L) we can consider the deadlockfree and live controls, respectively, which exist for the system described by the language L. By imp(L), fair(L), just(L) we have specified the controlled systems satisfying the fairness notions given in [18]: impartiality, fairness and justice. The control principle preg assigns all prefix closed regular sublanguages to a given language, thus, it describes all controls where the controlled system can be modelled by a finite transition system. Similarly, fnl(L) describes those controls for L where the controlled system can be modelled by a Petri net.

To make our intensions more apparent, we consider the problem of the five philosophers as given by the Petri net N in Fig. 1.

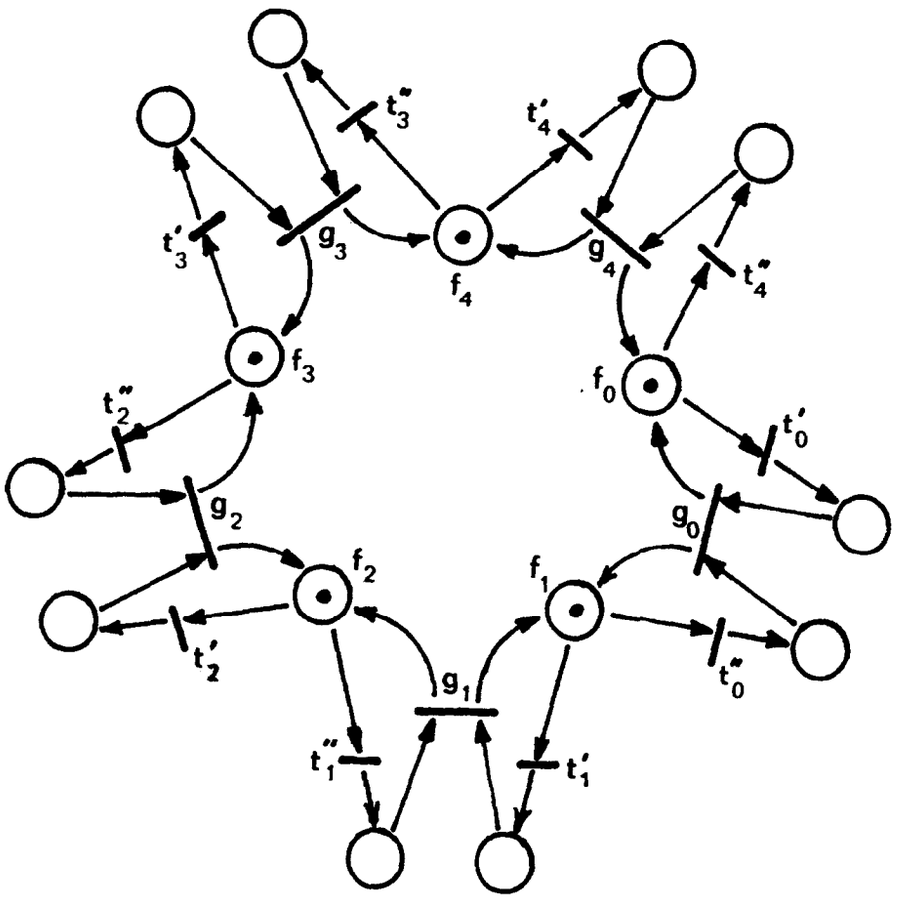

Fig. 1.

Deadlocks are possible in this net, e.g. after the execution of the sequence $u = t'_0t'_1t'_2t'_3t'_4$. This fact is expressed by $L_N \notin \mathrm{DFR}$.

As is well-known, a deadlock-free solution of the problem can be obtained by synchronizing the actions t'_i and t''_i for each $i = 0, \ldots, 4$. Hence we have $L_i \in \mathrm{dfr}(L_N)$ for

$$L_1 := L_N \cap (\overline{\{t'_0t''_0, t'_1t''_1, t'_2t''_2, t'_3t''_3, t'_4t''_4\}^*} \,|\, \{g_0, g_1, g_2, g_3, g_4\}^*),$$

and L_1 describes a possible deadlock-free solution (control) of the problem.

Another deadlock-free solution can be described by

$$L_2 := \overline{\{t_0' t_0'' g_0 t_1' t_1'' g_1 t_2' t_2'' g_2 t_3' t_3'' g_3 t_4' t_4'' g_4\}^*} \in \mathrm{dfr}(L_N).$$

This solution (being very restrictive) is even impartial ($L_2 \in \mathrm{imp}(L_N)$) while the solution given by L_1 was not.

Obviously, there are many different deadlock-free solutions under the aspects of control (i.e. by restrictions of the possibilities) for the problem of the five philosophers. We can study these solutions by the family $\mathrm{dfr}(L_N)$. In the same way we can study fair solutions by the family $\mathrm{fair}(L_N)$, etc.

We are then able to investigate:

(a) Combinations of different properties by families $c(L_N) \cap c'(L_N)$ (see Definition 3.2 below), e.g., $\mathrm{fair}(L_N) \cap \mathrm{dfr}(L_N)$ describes the fair and deadlock-free solutions.

(b) Comparisons of controls for a given property: There exists a special (least restrictive) deadlock-free solution of the problem such that all other deadlock-free solutions are more restricting the possibilities of the philosophers than this special solution. A related result does not hold for the fair solutions (see Section 6).

(c) Comparisons of controls of different types: Fair solutions need not be live and vice versa.

Corollary 1.5

(1) $L \in \mathrm{DFR}$ *iff* $\forall u \in L\ \exists w \in \mathrm{Adh}(L)$: $u \sqsubseteq w$.

(2) $L \in \mathrm{LIVE}$ *iff* $\forall u \in L\ \exists w \in \mathrm{Adh}(L)$: $u \sqsubseteq w \wedge \pi_w = \boldsymbol{\omega}$.

(3) $L \in \mathrm{PFIN}$ *iff* $\mathrm{Adh}(L) = \emptyset$.

Proposition 1.6. *If* $L \in \mathrm{PREG} \cup \mathrm{FNL}$, *then we have*:

(1) $L \in \mathrm{DFR}$ *iff* $\forall u \in L\ \exists u', v \in T^*$: $uu'v^\omega \in \mathrm{Adh}(L)$,

(2) $L \in \mathrm{LIVE}$ *iff* $\forall u \in L\ \exists u', v \in T^*$: $uu'v^\omega \in \mathrm{Adh}(L) \wedge \pi_v \geqslant \mathbf{1}$.

Proof. We only prove (2), the proof of (1) being similar. If $L \in \mathrm{LIVE}$, then there exists a $w \in \mathrm{Adh}(L)$ with $u \sqsubseteq w$ and $\pi_w = \boldsymbol{\omega}$ (by Corollary 1.5(2)). We decompose $w = uu_1u_2 \ldots$ such that $\pi_{u_i} \geqslant \mathbf{1}$ for all $i = 1, 2, 3, \ldots$.

If $L = L_S$ for the finite transition system $S = (T, Q, f, q_0)$ (in the case $L \in \mathrm{PREG}$), then we consider a state sequence $q_1, q_2, q_3, \ldots$ where $q_1 \in f(q_0, u)$ and $q_{i+1} \in f(q_i, u_i)$ for all $i = 1, 2, 3, \ldots$. Since Q is finite, we have some $j > i > 0$ with $q_j = q_i$. If $L = L_N$ for the Petri net $N = (P, T, F, V, m_0)$ (in the case FNL), then we consider the sequence of the markings $m_1, m_2, m_3, \ldots$ where $m_i := m_0 + \Delta u u_1 \ldots u_i$. We have some $j > i > 0$ with $m_j \geqslant m_i$.

In both cases, we choose $u' := u_1 \ldots u_i$, $v := u_{i+1} \ldots u_j$ (note that $\Delta v \geqslant \mathbf{0}$ in the case of the Petri nets) and then we have $uu'v^\omega \in \mathrm{Adh}(L)$ and $\pi_v \geqslant \mathbf{1}$.

It is trivial that the given condition implies $L \in \mathrm{LIVE}$. □

2. Decidability of liveness and fairness conditions

Theorem 2.1. *The problems* "$L \in$ DFR (LIVE, IMP, FAIR, JUST)?" *are decidable for languages* $L \in \mathrm{PREG} \cup \mathrm{FNL}$.

The decidability for the case $L \in \mathrm{PREG}$ was proved in [10].

Without loss of generality we can assume that L is given by $L = L_S$ where $S = (T, Q, f, q_0)$ is a finite (initially connected) transition system with $\mathrm{card}(f(q, t)) \leq 1$ for all $q \in Q$, $t \in T$ (i.e. S works deterministically). Then we have:

$$L \in \mathrm{DFR} \quad \text{iff} \quad \forall q \in Q\, \exists t \in T : f(q, t) \neq \emptyset,$$

$$L \in \mathrm{LIVE} \quad \text{iff} \quad \forall q \in Q\, \forall t \in T\, \exists u \in T^* : |u| < \mathrm{card}(Q) \wedge f(q, ut) \neq \emptyset,$$

$$L \notin \mathrm{IMP} \quad \text{iff} \quad L_0^{\mathrm{imp}} \neq \emptyset,$$

where

$$L_0^{\mathrm{imp}} := \{v \mid \exists q \in Q : q \in f(q, v) \wedge \pi_v \not\geq \mathbf{1}\},$$

$$L \notin \mathrm{FAIR} \quad \text{iff} \quad L_0^{\mathrm{fair}} \neq \emptyset,$$

where

$$L_0^{\mathrm{fair}} := \{v \mid \exists q \in Q : q \in f(q, v) \wedge \exists t \in T : \pi_v(t) = 0 \wedge f(q, t) \neq \emptyset\},$$

$$L \notin \mathrm{JUST} \quad \text{iff} \quad L_0^{\mathrm{just}} \neq \emptyset,$$

where

$$L_0^{\mathrm{just}} := \{v \mid \exists q \in Q : q \in f(q, v) \wedge \exists t \in T\, \forall v' \sqsubseteq v : \pi_v(t) = 0 \wedge f(q, v't) \neq \emptyset\}.$$

Obviously, the languages L_0^{imp}, L_0^{fair}, and L_0^{just} are regular and all the conditions are decidable.

Now, let L be the language L_N of the firing sequences of a Petri net $N = (P, T, F, V, m_0)$. The problems "$L \in$ DFR?" and "$L \in$ LIVE?" are decidable since the reachability problem is decidable [17]: We have $L_N \notin$ DFR iff a dead marking is reachable in N, and $L_N \in$ LIVE iff the initial marking m_0 is live (the equivalence to the reachability problem was shown in [15]).

The further proof uses the coverability tree $\tau_N = (S', E, \mu)$ of the Petri net N, which can be defined as follows [6]:

(i) $S' \subseteq T^*$ is the set of nodes,

(ii) $\mu : S' \to (\mathbb{N} \cup \{\omega\})^P$ is the node labelling function,

(iii) $E := \{(r, rt) \mid r \in T^* \wedge t \in T \wedge r, rt \in S'\}$ is the set of directed edges,

(iv) S' and μ are defined recursively:

(0) $e \in S'$, $\mu(e) := m_0$ (e is the root of the tree),

(1) If $r \in S'$ and $\mu(r) \neq \mu(s)$ for all proper prefixes s of r, then $rt \in S'$ for all $t \in T$ with $t^- \leq \mu(r)$.

For these rt the function μ_r is defined by

$$\mu(rt)(p) := \begin{cases} \omega & \text{if } \exists s \sqsubseteq r\colon \mu(s) \leqslant \mu(r) + \Delta t \wedge \\ & \qquad \wedge\, \mu(s)(p) < (\mu(r) + \Delta t)(p), \\ (\mu(r) + \Delta t)(p) & \text{otherwise (whereby } \omega + n = \omega). \end{cases}$$

(2) No other r are in S'.

Some leaves of the tree are called *loop ends*, they are given by

$$S_1 := \{r \mid r \in S' \wedge \exists s \sqsubsetneq r\colon \mu(r) = \mu(s)\}.$$

The corresponding nodes with the identical labels are called *loop starts* and are given by

$$S_0 := \{s \mid s \in S' \wedge \exists r\colon s \sqsubsetneq r\colon \mu(r) = \mu(s)\}.$$

By the identification of the loop ends with their corresponding loop starts we get the transition system $S(N) = (T, S, f, e)$ where

$$S := S' \setminus S_1,$$

$$f(r, t) := \begin{cases} \{rt\} & \text{if } rt \in S, \\ \{s\} & \text{if } rt \in S_1 \wedge s \sqsubseteq r \wedge \mu(rt) = \mu(s), \\ \emptyset & \text{otherwise.} \end{cases}$$

$S(N)$ is finite (since τ_N is finite) and deterministic, i.e. $\operatorname{card}(f(r, t)) \leqslant 1$ for all $r \in S$, $t \in T$.

Lemma 2.2. (1) $L_N \subseteq L_{S(N)}$.

(2) *If $f(s, v) = \{s'\}$ in $S(N)$ and $\mu(s)(p) = \omega$ iff $\mu(s')(p) = \omega$ for all $p \in P$, then a sequence $u \in T^*$ with $uv \in L_N$ can be constructed.*

Proof. (1) corresponds to a well-known property of τ_N (cf. Lemma (2) in [6]).

To prove (2), we know that v is firable with respect to the finite coordinates of $\mu(s)$ (by the construction of $S(N)$)). By Lemma (3) in [6], we can construct a sequence $u \in L_N$ with

$$(m_0 + \Delta u)(p) \begin{cases} = \mu(s)(p) & \text{if } \mu(s)(p) \neq \omega, \\ \geqslant m(p) & \text{if } \mu(s)(p) = \omega, \end{cases}$$

where m is an arbitrarily chosen marking. If we choose m sufficiently large, we can construct u such that v is firable in $m_0 + \Delta u$, i.e. $uv \in L_N$. □

Lemma 2.3. *The condition*

$$(*) \qquad \exists u \in T^*\, \exists v \in L_0\colon uv^\omega \in \operatorname{Adh}(L)$$

is decidable for regular languages L_0 and languages $L \in$ FNL. Such sequences u, v can be constructed if they exist.

Proof. We suppose $L = L_N$ for $N = (P, T, F, V, m_0)$ and consider the corresponding transition system $S(N) = (T, S, f, e)$. We show that the condition (*) is equivalent to the condition

$$(**) \quad \exists s \in S\, \exists v \in L_0 : f(s, v) \neq \emptyset \wedge \Delta v \geq \mathbf{0}.$$

If $uv^\omega \in \mathrm{Adh}(L_N)$, then we have $\Delta v \geq \mathbf{0}$, and, by Lemma 2.2(1), we have $f(s, v) \neq \emptyset$ where $f(e, u) = \{s\}$. Hence, (*) implies (**). If $f(s, v) \neq \emptyset$ and $\Delta v \geq \mathbf{0}$, then we have $f(s, v^i) \neq \emptyset$ for all $i \in \mathbb{N}$. There must exist a number i such that it holds for $f(s, v^i) = \{s'\}$, $f(s', v) = \{s''\}$:

$$\forall p \in P : \mu(s') = \omega \leftrightarrow \mu(s'')(p) = \omega$$

(since the number of ω-coordinates cannot decrease if we go futher in $S(N)$). Thus, we can apply Lemma 2.2(2) and find a sequence u such that $uv \in L_N$. It follows that $uv^\omega \in \mathrm{Adh}(L_N)$ since $\Delta v \geq \mathbf{0}$. The next step is to show that (**) is decidable, i.e. the problem "$L_1 \neq \emptyset$?" is decidable for

$$L_1 := \{v \mid v \in L_0 \wedge \Delta v \geq \mathbf{0} \wedge \exists s \in S : f(s, v) \neq \emptyset\}.$$

By $\pi(L) := \{\pi(u) \mid u \in L\}$ we denote the set of the Parikh-vectors corresponding to a language L. Then we have $L_1 = L_0 \cap L_2 \cap \bigcup_{s \in S} L_s$, i.e. $\pi(L_1) = \pi(L_0) \cap \pi(L_2) \cap \bigcup_{s \in S} \pi(L_s)$, where $L_2 = \{v \mid \Delta v \geq \mathbf{0}\}$ and $L_s = \{v \mid f(s, v) \neq \emptyset\}$.

Obviously, the sets $\pi(L_0)$, $\pi(L_2)$, $\pi(L_s)$ are all computable semilinear sets (L_0, L_s are regular, $\pi(L_2)$ contains the nonnegative integer solutions of a system of linear inequalities with integer coefficients). Hence, $\pi(L_1)$ is a computable semilinear set and "$\pi(L_1) \neq \emptyset$?" is decidable.

In conclusion: Condition (*) holds if we can start with a sequence $v \in L_0$ in some state s of $S(N)$, whereby $\Delta v \geq \mathbf{0}$. □

Remark. It can be shown that condition (*) holds iff

$$\exists s \in S_0\, \exists v \in L_0' : f(s, v) = \{s\} \wedge \Delta v \geq \mathbf{0},$$

whereby $L_0' := \{v''v^i v' \mid i \in \mathbb{N} \wedge v'v'' = v \in L_0\}$ (since each infinite path through $S(N)$ has to pass through some loop start infinitely often). Thus it suffices to look for cyclic paths through the loop starts in $S(N)$ which are labelled by a sequence $v \in L_0'$ with $\Delta v \geq \mathbf{0}$.

Now we continue the proof Theorem 2.1 by the application of Lemma 2.3 to the cases IMP, FAIR, JUST, where $L \in \mathrm{FNL}$, i.e. $L = L_N$ for $N = (P, T, F, V, m_0)$.

We have $L_N \notin \mathrm{IMP}$ iff there exists an infinite sequence $w = t_1 t_2 t_3 \ldots$ in $\mathrm{Adh}(L_N)$ with $\pi_w \neq \boldsymbol{\omega}$, i.e. $t_i \neq t$ for some $t \in T$ and all i that are greater than some $k \in \mathbb{N}$. There must exist $i, j \in \mathbb{N}$ with $j > i \geq k$ such that $\Delta t_{i+1} \ldots t_j \geq \mathbf{0}$ (since there is an infinite non-decreasing subsequence in $(m_0 + \Delta t_1 \ldots t_i)_{i \in \mathbb{N}}$). Hence there exist $u := t_1 \ldots t_i$, $v := t_{i+1} \ldots t_j$ such that $uv^\omega \in \mathrm{Adh}(L_N)$ and $\pi_v(t) = 0$. Since the reverse is

trivial, we have:

$$L_N \notin \text{IMP} \text{ iff } \exists u, v \in T^*: uv^\omega \in \text{Adh}(L_N) \wedge \pi_v \not\geqslant \mathbf{1}, \text{ i.e.}$$

$$\text{iff } \exists u \in T^* \exists v \in L_0: uv^\omega \in \text{Adh}(L_N),$$

where $L_0 := \{v \mid \pi_v \not\geqslant \mathbf{1}\}$ is regular.

The last condition is decidable by Lemma 2.3.

We have $L_N \notin \text{FAIR}$ iff there exists an infinite sequence $w = u_0 u_1 u_2 \ldots (u_i \in T^*)$ in $\text{Adh}(L_N)$ and some $t \in T$ such that $u_0 \ldots u_i t \in L_N$ for all $i \in \mathbb{N}$ and $\pi_w(t) \neq \omega$, i.e. we can assume $\pi_{u_i}(t) = 0$ for all $i \geqslant 1$. Using the same argument as in the proof concerning IMP, we can find $j > i \geqslant 1$ with $\Delta u_{i+1} \ldots u_j \geqslant \mathbf{0}$, and then we can show (with $u := u_0 \ldots u_i$, $v := u_{i+1} \ldots u_j$):

$$L_N \notin \text{FAIR} \text{ iff } \exists u, v \in T^* \exists t \in T: uv^\omega \in \text{Adh}(L_N)$$

$$\wedge \pi_v(t) = 0 \wedge \forall i \in \mathbb{N}: uv^i t \in L_N.$$

The last condition implies (by Lemma 2.2(1) and with regard to the corresponding transition system $S(N) = (T, S, f, e)$):

$$\exists u \in T^* \exists v \in L_0: uv^\omega \in \text{Adh}(L_N),$$

where

$$L_0 := \{v \mid \exists t \in T \exists s \in S: \pi_v(t) = 0 \wedge f(s, t) \neq \emptyset \wedge f(s, v) \neq \emptyset\}.$$

We show that both conditions are equivalent: the crucial point is that $uv^\omega \in \text{Adh}(L_N)$, $v \in L_0$, does not imply $uv^i t \in L_N$ for any $i \in \mathbb{N}$. We must step back to the application of Lemma 2.2(2) as in the proof of Lemma 2.3. By $uv^\omega \in \text{Adh}(L_N)$ we have $\Delta v \geqslant \mathbf{0}$ and there exists some number i such that $\mu(f(s, v^i))$ and $\mu(f(s, v^{i+1}))$ coincide with respect to their ω-coordinates. Now we can apply Lemma 2.3, but we choose the marking m (cf. the proof of Lemma 2.2(2)) so large that we get a sequence u' such that both v and t are firable in $m_0 + \Delta u'$, i.e. $u'v$, $u't \in L_N$ for all $i \in \mathbb{N}$.

This proves that $L_N \notin \text{FAIR}$ holds iff there exist $u \in T^*$, $v \in L_0$ with $uv^\omega \in \text{Adh}(L_N)$. This condition is decidable by Lemma 2.3 since

$$L_0 = \bigcup_{t \in T} \left[\{v \mid \pi_v(t) = 0\} \cap \bigcup_{s \in S, f(s,t) \neq \emptyset} \{v \mid f(s, v) \neq \emptyset\} \right]$$

is regular.

Finally, we have $L_N \notin \text{JUST}$ iff there exists an infinite sequence $w = t_1 t_2 t_3 \ldots$ in $\text{Adh}(L_N)$, a transition $t \in T$ and some $k \in \mathbb{N}$ such that $t_i \neq t$ and $t_1 \ldots t_i t \in L_N$ for all $i > k$. Similar to the preceding proofs we can show:

$$L_N \notin \text{JUST} \text{ iff } \exists u, v \in T^* \exists t \in T: uv^\omega \in \text{Adh}(L_N)$$

$$\wedge \pi_v(t) = 0 \wedge \forall i \in \mathbb{N} \forall v' \sqsubseteq v: uv^i v' t \in L_N.$$

Furthermore, we have

$$L_N \notin \text{JUST} \text{ iff } \exists u \in T^* \exists v \in L_0: uv^\omega \in \text{Adh}(L_N),$$

where

$$L_0 := \{v \mid \exists t \in T\, \exists s \in S\colon \pi_v(t) = 0 \wedge f(s, v) \neq \emptyset \wedge \forall v' \sqsubseteq v\colon f(s, v't) \neq \emptyset\}.$$

The proof is similar to the analogous proof for fairness, but now we have to choose the marking m so large that v and all sequences $v't$ for $v' \sqsubseteq v$ become firable in $m_0 + \Delta u'$.

$$L_0 = \bigcup_{t \in T} \left[\{v \mid \pi_v(t) = 0\} \cap \bigcup_{s \in S} \{v \mid f(s, v) \neq \emptyset \wedge \forall v' \sqsubseteq v\colon f(s, v't) \neq \emptyset\} \right]$$

is regular and thus Lemma 2.3 is applicable. □

Remark. One can show by similar arguments as given, concerning the remark after Lemma 2.3 that

$$L \notin \mathrm{IMP} \quad \text{iff} \quad \exists v \in L_0^{\mathrm{imp}}\colon \Delta v \geq \mathbf{0},$$

$$L \notin \mathrm{FAIR} \quad \text{iff} \quad \exists v \in L_0^{\mathrm{fair}}\colon \Delta v \geq \mathbf{0},$$

$$L \notin \mathrm{JUST} \quad \text{iff} \quad \exists v \in L_0^{\mathrm{just}}\colon \Delta v \geq \mathbf{0},$$

where the languages L_0^{imp}, L_0^{fair}, L_0^{just} are defined for $\boldsymbol{S}(\boldsymbol{N})$ in the same way as for $\boldsymbol{S}$ in the proof part "$L \in \mathrm{PREG}$".

3. Relations between live and fair controls

The properties corresponding to extensional control principles are properties of the controlled systems only, they do not depend on the original uncontrolled systems. The language $\{a\}^*$ is contained in $\mathrm{fair}(\{a\}^*)$, but also in $\mathrm{fair}(\{a, b\}^*)$, $\mathrm{fair}(\{a, b, c\}^*)$, etc. The definition of $\mathrm{fair}(L)$ may be insufficient to meet the intensions of a fair control: We have $\{a\}^* \in \mathrm{fair}(\{a, b\}^*)$ although b is infinitely often enabled by a^ω with respect to the uncontrolled system given by $\{a, b\}^*$. Hence it might be better to consider 'relative fairness' and 'relative justice' with respect to the uncontrolled system (given by L).

Definition 3.1

$$L' \in \mathrm{rfair}(L) \quad \text{iff} \quad \forall t \in T\, \forall w \in \mathrm{Adh}(L')\colon (\exists^\infty u \sqsubseteq w\colon ut \in L) \to \pi_w(t) = \omega.$$

$$L' \in \mathrm{rjust}(L) \quad \text{iff} \quad \forall t \in T\, \forall w \in \mathrm{Adh}(L')\colon (\forall^\infty u \sqsubseteq w\colon ut \in L) \to \pi_w(t) = \omega.$$

Remark. Referring to a fixed language L (i.e. to a special given uncontrolled system), there exists the possibility to define an extensional control principle by $C := \mathrm{rfair}(L)$ ($\mathrm{rjust}(L)$). Advantages may then arise by the application of properties of extensional control principles (cf. Section 5).

In the sequel we sometimes refer to imp, fair, just, rfair, rjust shortly as the fairness control principles. Since some fairness can be enforced by restrictions to finite languages, it is interesting to study those controls which are fair and deadlock-free or which are fair and non-blocking (cf. Definition 3.5 below).

Thus, we are interested in the study of controls which realize several properties. We are also interested in the preservation of the properties of a control principle c by another control principle c' (for example: all live controls are deadlock-free). This leads to the following definition:

Definition 3.2. The *conjunction* $c\&c'$ of two control principles c, c' is defined by

$$c\&c'(L) := c(L) \cap c'(L) \quad \text{for all } L \in \mathrm{CONT}.$$

The control principle c is *covered* by c', $c \leqslant c'$, iff we have

$$c(L) \subseteqq c'(L) \quad \text{for all } L \in \mathrm{CONT}.$$

Obviously, $c\&c' \leqslant c, c'$, i.e. the properties realized by c and c' are preserved by $c\&c'$. We also have the following corollary.

Corollary 3.3. *If* c_1, c_2 *are extensional, then we have:*

(1) $c_1 \leqslant c_2$ *iff* $C_1 \subseteq C_2$,

(2) $c := c_1\&c_2$ *is extensional with* $C = C_1 \cap C_2$.

Theorem 3.4. *For* $\mathrm{card}(T) \geqslant 3$ *we have the relations between the control principles* dfr, live, imp, fair, just, pfin, rfair, *and* rjust *as represented by Fig.* 2. (*It also represents the relations between the families* DFR, LIVE, IMP, FAIR, JUST, PFIN.)

Proof. It immediately follows from the definitions that

$$\mathrm{live} \leqslant \mathrm{dfr},$$

$$\mathrm{pfin} \leqslant \mathrm{imp} \leqslant \mathrm{rfair} \leqslant \mathrm{fair} \leqslant \mathrm{just},$$

$$\mathrm{rfair} \leqslant \mathrm{rjust} \leqslant \mathrm{just}.$$

Furthermore, for $u \in L \in \mathrm{DFR} \cap \mathrm{IMP}$ there exists some $w \in \mathrm{Adh}(L)$ with $u \sqsubseteq w$ (by Corollary 1.5(1)) and $\pi_w = \omega$ (by the definition of IMP). This implies $L \in \mathrm{LIVE}$ by Corollary 1.5(2). Thus, we have $\mathrm{DFR} \cap \mathrm{IMP} \subseteqq \mathrm{LIVE}$, i.e. $\mathrm{dfr}\&\mathrm{imp} \leqslant \mathrm{live}$.

It remains to show the inequalities. Since it can be proved that $\mathrm{live}\&\mathrm{imp} = \mathrm{live}\&\mathrm{rfair} = \mathrm{live}\&\mathrm{fair}$ holds for $\mathrm{card}(T) = 2$, we consider the alphabet $T = \{a, b, c\}$ concerning 2″–4″ (the numbers refer to the position in Fig. 2):

2″: $L := \overleftarrow{\{ab\}^* \cdot \{aabc\}^*} \in (\mathrm{LIVE} \cap \mathrm{FAIR}) \setminus \mathrm{IMP}$,
hence $L \in \mathrm{live}\&\mathrm{rfair}(L)$ (note that $L \in \mathrm{rfair}(L)$ iff $L \in \mathrm{FAIR}$)
and $L \notin \mathrm{imp}(L)$, i.e. $\mathrm{live}\&\mathrm{rfair} \not\leqslant \mathrm{imp}$.

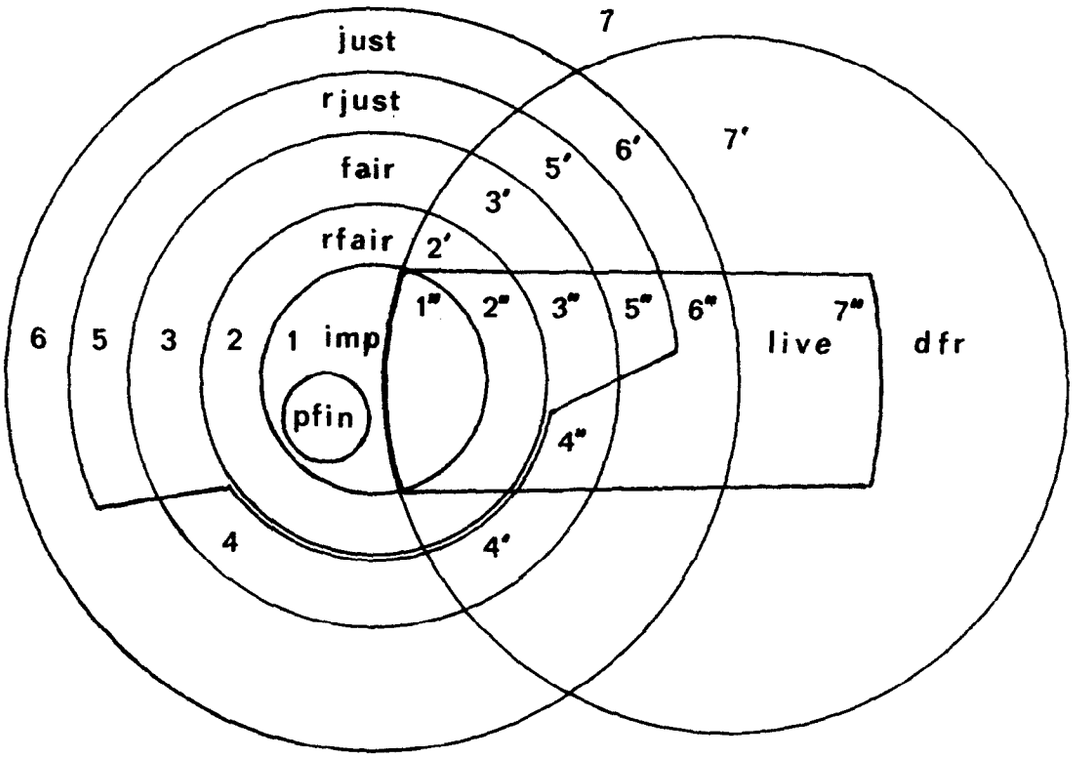

Fig. 2.

3″: $L := \overleftarrow{\{ab\}^* \cdot \{c\}^* \cdot \{aabc\}^*}$, $L' := \overleftarrow{\{ab\}^* \cdot \{aabc\}^*}$,
with $L' \in$ live&fair&rjust(L) and $L' \notin$ rfair(L)
(consider $(ab)^\omega$), i.e. live&fair&rjust $\not\leq$ rfair.

4″: $L := \{a, b, c\}^*$, $L' := \overleftarrow{\{ab\}^* \cdot \{aabc\}^*}$,
with $L' \in$ live&fair$(L)\backslash$rjust(L), i.e. live&fair $\not\leq$ rjust.

It suffices to consider $T = \{a, b\}$ for the remainder:

5″: $L := \overleftarrow{\{aa\}^* \cdot \{ab\}^*} \in (\text{LIVE} \cap \text{JUST})\backslash\text{FAIR}$,
hence $L \in$ live&rjust$(L)\backslash$fair(L), i.e. live&rjust $\not\leq$ fair.

6″: live&just $\not\leq$ fair by the example given for 5″,
$L := \{a, b\}^*$, $L' := \overleftarrow{\{aa\}^* \cdot \{ab\}^*}$,
with $L' \in$ live&just$(L)\backslash$rjust(L), i.e. live&just $\not\leq$ rjust.

7″: $\{a, b\}^* \in$ LIVE$\backslash$JUST, hence live $\not\leq$ just.

2′: $L := \{a\}^* \in (\text{DFR} \cap \text{FAIR})\backslash(\text{LIVE} \cup \text{IMP})$,
hence $L \in$ dfr&rfair(L), i.e. dfr&rfair $\not\leq$ live, imp.

3′: $L := \overleftarrow{\{aa\}^* \cdot \{b\}}$, $L' := \{a\}^*$,
with $L' \in$ dfr&fair&rjust(L), $L' \notin$ live$(L) \cup$ rfair(L),
hence dfr&fair&rjust $\not\leq$ live, rfair.

4′: $L := \{a, b\}^*$, $L' := \{a\}^*$,
with $L' \in$ dfr&fair(L), $L' \notin$ live$(L) \cup$ rjust(L),
hence dfr&fair $\not\leq$ live, rjust.

5′: $L := \overleftarrow{\{aa\}^* \cdot \{b\}^*} \in (\mathrm{DFR} \cap \mathrm{JUST}) \setminus (\mathrm{LIVE} \cup \mathrm{FAIR})$,
hence dfr&just $\not\leq$ live, fair.

6′: For the example of 4′ we also have $L' \in \mathrm{dfr\&just}(L)$,
hence dfr&just $\not\leq$ live, rjust.

7′: $L := \overleftarrow{\{a\}^* \cdot \{b\}^*} \in \mathrm{DFR} \setminus (\mathrm{LIVE} \cap \mathrm{JUST})$, hence dfr $\not\leq$ live, just.

2-7: Examples can be constructed from the corresponding languages for 2′-7′: we can consider $L \cup \{b, ba\}$ and $L' \cup \{b, ba\}$, for example, such that the new languages are not contained in DFR. □

Remark. The theorem concerns the consideration of all languages from CONT. If we restrict the consideration to special classes of languages, then some of the control principles may coincide.

Further (non-extensional) control principles of interest are the control principles 'conflict resolution' and 'non-blocking':

Definition 3.5

$$L' \in \mathrm{crs}(L) \text{ iff } \forall ut, ut' \in L' : t, t' \in T \wedge t \neq t' \rightarrow utt' \in L.$$

$$L' \in \mathrm{nbl}(L) \text{ iff } \forall u \in L' : (\exists t \in T : ut \in L) \rightarrow (\exists t' \in T : ut' \in L').$$

We remark that our notion of conflict resolution concerns only (binary) conflicts where one action can loose its concession by performing another action: if such a case appears in L, then the conflict resolving control has to decide which one of the conflicting actions can be performed. It is a disadvantage of the non-deterministic interleaving by the consideration of abstract languages $L \subseteq T^*$ that it does not allow to study the control with respect to conflicts as in the Petri net examples of Fig. 3.

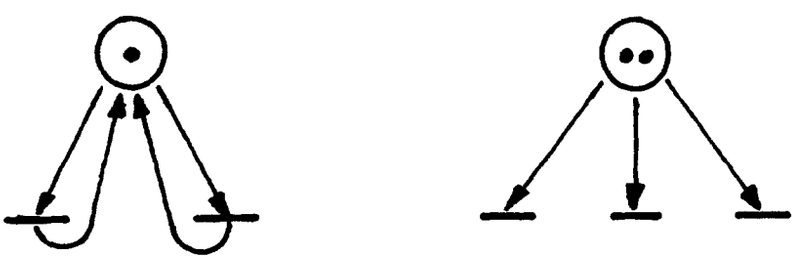

Fig. 3.

The consideration of languages over the alphabet $(P(T))^*$ can be helpful to study such conflicts (cf. [7, 8] and the example of Fig. 5 in this paper).

We furthermore remark that our conflict resolution concerns only the conflicts in the uncontrolled system, the controlled system may have new conflicts: we have, for example, $\overleftarrow{\{a, b\}} \in \mathrm{crs}(\overleftarrow{\{ab, ba\}})$. This would be excluded if we considered the

extensional control principle 'persistency' given by

$$L \in \text{PERS} \text{ iff } \forall ut, ut' \in L\colon t, t' \in T \wedge t \neq t' \to utt' \in L.$$

By the notion of non-blocking we exclude termination by control when the uncontrolled system can work further. The following corollary holds.

Corollary 3.6

(1) $\text{dfr} \leqslant \text{nbl}$.

(2) $L \in \text{DFR}$ *iff* $\text{nbl}(L) = \text{dfr}(L)$.

(3) $L \in \text{DFR}$ *iff* $\text{nbl\&pfin}(L) = \emptyset$.

Theorem 3.7. *Suppose* $c \in \{\text{cont, dfr, live, imp, fair, just, pfin, preg}\}$. *Then the problems* "$L' \in c(L)$?" *are decidable for* L, $L' \in \text{PREG}$ *and* L, $L' \in \text{FNL}$, *respectively.*

Proof. We have $L' \in \text{cont}(L)$ iff $L' \subseteq L$, and the inclusion problems are decidable for languages from PREG and FNL, respectively (for FNL by reduction to the liveness problem [16]). For the extensional control principles c we have $L' \in c(L)$ iff $L' \in \text{cont}(L) \wedge L' \in C$ for the corresponding sets C. Hence, the results for $c = \text{dfr}$, live, imp, fair, just are consequences of Theorem 2.1.

In the same way, "$L' \in \text{pfin}(L)$?" is decidable since "$L' \in \text{PFIN}$?" is decidable (in the case $L_N \in \text{FNL}$ we have $L_N \in \text{PFIN}$ iff $L_{S(N)} \in \text{PFIN}$, where $S(N)$ is the finite transition system constructed in the proof of Theorem 2.1. The decidability of "$L' \in \text{PREG}$?" for $L' \in \text{FNL}$ was shown in [21]. (*Remark*: The problem "$L' \in \text{FNL}$?" for $L' \in \text{PREG}$ is an open problem.) □

Some further results have been shown in [11]: Theorem 3.7 also holds

(a) for L, $L' \in \text{PREG} \cup \text{FNL}$ (by proving the decidability of the inclusion problem for languages L, $L' \in \text{PREG} \cup \text{FNL}$),

(b) for $c \in \{\text{rfair, rjust, crs, nbl}\}$.

4. Controls by finite automata

Since control devices often work as finite automata we consider finite non-deterministic automata

$$A = (P(T), T, Z, h, z_0)$$

as *control automata*, where $P(T)$, T, Z are the finite nonempty sets of inputs, outputs and states, respectively, $z_0 \in Z$ is the initial state and $h\colon Z \times P(T) \to P(T \times Z)$ is the non-deterministic output/next-state function with

$$h(z, \emptyset) = \emptyset,$$

$$\emptyset \neq \{t \mid \exists z'\colon (t, z') \in h(z, U)\} \subseteq U \quad \text{for all } z \in Z,\ U \in P(T) \setminus \{\emptyset\}.$$

(The last condition ensures that the control by automata works non-blocking.)

The control automaton A and the system to be controlled form an interactive system: the automaton A receives as input the set U of all actions from T which could be performed in the next step by the system and decides by its output $t \in U$ which action can be performed. A can be considered as an R-robot working in the environment L in the sense of [4]. Related controls of the internal behaviour are studied in [1]. Controls of Petri nets by control automata have been studied in [7, 8] and [12]. Obviously, the concept of control automata is powerful enough to modelize priority rules and fifo-queues, for example. But it may fail with respect to the control of concurrent work (realizing 'Max-Semantics' [19], for example). Again, this may lead to the examination of languages over the alphabet $P(T)$ (in this case the control automata may receive inputs from $P(P(T))$ giving information about concurrently performable sets of actions and it decides in favour of a set of actions to be performed concurrently by an output from $P(T)$, cf. [7, 8]).

We define the following according to our interpretation of control by automata.

Definition 4.1. Let $A = (P(T), T, Z, h, z_0)$ be a control automaton. The *result of the control of* $L \in \mathrm{CONT}$ *by* A is the language L/A with

$$e \in L/A,$$

$$t_1 \dots t_n \in L/A \text{ iff } \exists z_1, \dots, z_n \in Z \; \forall i = 0, \dots, n-1:$$

$$(t_{i+1}, z_{i+1}) \in h(z_i, \{t \mid t_1 \dots t_i t \in L\}).$$

For a nonempty class **aut** of control automata we define the *control principle* aut by

$$\mathrm{aut}(L) := \{L/A \mid A \in \mathbf{aut}\}.$$

Proposition 4.2. *The control principles* aut *are not extensional and it holds that* $\mathrm{aut} \leqslant \mathrm{nbl}$.

Proof. We have $\mathrm{aut} \leqslant \mathrm{nbl}$ since the control automata A work non-blocking by definition. The non-extensionality follows by:

Lemma 4.3. *If* $c \leqslant \mathrm{nbl}$ *and* $c(\mathrm{CONT}) \not\subseteq \mathrm{DFR}$, *then* c *is not monotonous* (*and hence not extensional by Corollary* 1.3(1)).

Proof. If c was monotonous, then $L \in c(\mathrm{CONT})$ would imply $L \in c(T^*)$. But, there exists a language $L \in c(\mathrm{CONT})$ with $L \notin \mathrm{DFR}$ and hence $L \notin \mathrm{nbl}(T^*)$, i.e. $L \notin c(T^*)$ since $c \leqslant \mathrm{nbl}$. □

To characterize the results of automata controls we have the following theorem.

Theorem 4.4. *Let* **aut1** *denote the class of all control automata. Then it holds*:

PREG = aut1(PREG),
FNL $\not\subseteq$ aut1(FNL) $\not\subseteq$ PREC = aut1(PREC) $\not\subseteq$ PREN $\not\subseteq$ aut1(PREN),
PREG *and* aut1(FNL) (*as well as* PREG *and* FNL) *are incomparable*

Proof. We have $L = L/A$ for $A = (P(T), T, \{z_0\}, h, z_0)$ with $h(z_0, U) := U \times \{z_0\}$ for all $U \in P(T)$. Thus, the following lemma holds.

Lemma 4.5. $C \subseteq \text{aut1}(C)$ *for all* $C \subseteq \text{CONT}$.

This and some intuitive arguments (using Church's thesis) prove all the left-to-right inclusions and also PREC = aut1(PREC). Now we consider the language L_S of a deterministic transition system $S = (T, Q, f, q_0)$ under the control of the control automaton $A = (P(T), T, Z, h, z_0)$. From S and A we define the transition system $S' := (T, Q \times Z, f', (q_0, z_0))$ with $L_{S'} = L_S/A$ in the following way:

For $q \in Q$, $z \in Z$, $t \in T$ we put

$$f'((q, z), t') := \{(q', z') \mid q' \in f(q, t') \wedge (t', z') \in h(z, \{t \mid f(q, t) \neq \emptyset\})\}.$$

To satisfy our convention that transition systems are initially connected, we can restrict the state set to the set $Q' := \bigcup_{u \in T^*} f'((q_0, z_0), u) \subseteq Q \times Z$ and restrict f' with respect to $Q' \times T$ and obtain $S/A := (T, Q', f', q_0)$. The verification of $L_{S/A} = L_S/A$ is left to the reader. Since S/A is finite if S is finite, we have aut1(PREG) $\subseteq$ PREG.

It remains to show the following inequalities:

- FNL $\neq$ aut1(FNL):
  $L := \overline{\{abbaa\}} \in$ aut1(FNL) (easy to verify), but $L \notin$ FNL (since $abbaa \in L_N$ implies $m_0 \geq a^-$, $m_0 + \Delta abba \geq a^-$, by addition: $2m_0 + 2\Delta ab \geq 2a^-$, i.e. $m_0 + \Delta ab \geq a^-$ and hence $aba \in L_N$).
- aut1(FNL) $\neq$ PREC since even PREG $\not\subseteq$ aut1(FNL).
- PREN $\neq$ aut1(PREN):
  Let M be a recursively enumerable but not recursive subset of $\mathbb{N}$ and $L := \{t_1 \ldots t_n \mid n \in \mathbb{N} \wedge \forall i = 1, \ldots, n$: **if** $i \notin M$ **then** $t_i = a$ **else** $t_i \in \{a, b\}\}$.
  Then, for $A := (P(\{a, b\}), \{a, b\}, \{z_0\}, h, z_0)$ with $h(z_0, \{a\}) = \{(a, z_0)\}$, $h(z_0, \{a, b\}) = \{(b, z_0)\}$, we have $L/A = \tilde{w}$ with $w(i) :=$ **if** $i \notin M$ **then** a **else** b.
  Thus, $L \in$ PREN but $L/A \notin$ PREN (otherwise M would be recursive).
- PREG $\not\subseteq$ aut1(FNL):
  $L := \overline{\{b\} \cdot \{a\}^*} \cup \{a\} \in$ PREG, but $L \notin$ aut1(FNL), since $L = L_N/A$ for some control automaton A and some Petri net N would imply $\{a\}^* \subseteq L_N$ (note that $L \subseteq L_N$ implies $\Delta a \geq \mathbf{0}$), and this would imply $L \notin \text{nbl}(L_N)$, i.e. $L \notin \text{aut1}(L_N)$ by Proposition 4.2.
- aut1(FNL) $\not\subseteq$ PREG since even FNL $\not\subseteq$ PREG (well-known). □

Remark. c(PREG) $\not\subseteq$ PREN holds for c = dfr, live, imp, fair, just, rfair, rjust, crs, nbl. We can consider, for example, $L := \tilde{w}$ with $w(i) =$ **if** $i \in M$ **then** a **else** b, such that $L \in c(\{a, b\}^*)$ but $L \notin$ PREN if M is a not recursively enumerable subset of $\mathbb{N}$.

The following proposition is useful for the realization of controls by automata studied in Section 7.

Proposition 4.6. *Let* **aut1** *denote the class of all control automata. Then it holds that*

$$\text{preg\&nbl} \leqslant \text{aut1}.$$

Proof. Let $L' \in \text{preg\&nbl}(L)$ be given by $L' = L_S$, where $S = (T, Q, f, q_0)$ is a finite deterministic transition system. Then we have $L' = L/A$ for $A := (P(T), T, Q, h, q_0)$ with

$$h(q, U) = \begin{cases} \{(t, q') \mid t \in U \wedge q' \in f(q, t)\} & \text{if this set is not empty,} \\ U \times \{q\} & \text{otherwise,} \end{cases}$$

for $q \in Q$, $U \in P(T)$. □

5. Stepwise refinements of controls

Definition 5.1. The *superposition* $c * c'$ of two control principles c, c' is defined by $c * c'(L) := c(c'(L))$ for all $L \in \text{CONT}$.

The superposition of control principles reflects the stepwise construction of controls, e.g. aut∗crs∗live means first construction of a live system, then resolution of conflicts and finally realization by an automaton from **aut**. The goal is to obtain a control from aut&crs&live. But it turns out that such stepwise refinements need not result in controls satisfying the desired properties, for example: conflict resolution for a live system need not result in a live system (consider $\{a\}^* \in \text{crs}(\{a, b\}^*)$ showing that crs∗live ⩽ crs&live is not true). This observation is important for some analyzing methods too: the verification of some properties for an uncontrolled system is in general not relevant with respect to the controlled system (cf. [12] for the study of liveness and deadlock avoidance in Petri nets working under conflict resolving firing rules). The positive aspect of this observation is the possibility to control systems in order to satisfy properties which do not hold for the uncontrolled system.

Concerning the preservation of properties by stepwise refinements we use the following notions.

Definition 5.2. A control principle c is *left-invariant* iff $c' * c \leqslant c$ for all control principles c'.

A control principle c is *right-invariant* iff $c * c' \leqslant c$ for all control principles c'.

Corollary 5.3. (1) *c is left-invariant iff* $\text{cont} * c \leqslant c$ *iff* $\text{cont} * c = c$.

(2) *c is right-invariant iff* $c * \text{cont} \leqslant c$ *iff* $c * \text{cont} = c$ *iff c is monotonous.*

(3) *Extensional control principles are right-invariant.*

Proof. We prove the equivalences for right-invariant control principles; the corresponding proofs for left-invariant control principles are similar.

$c * c' \leqslant c$ for all c' implies $c * \mathrm{cont} \leqslant c$ (we choose $c' := \mathrm{cont}$), and $c * \mathrm{cont} \leqslant c$ implies $c * c' \leqslant c$ (since $c' \leqslant \mathrm{cont}$) for all c'.

Furthermore, it always holds that $c \leqslant c * \mathrm{cont}$.

For $L_1 \subseteq L_2$ we have $c(L_1) = c * \mathrm{cont}(L_1) \subseteq c * \mathrm{cont}(L_2) = c(L_2)$, showing that right-invariant control principles are monotonous.

On the other hand, if c is monotonous, then we have $c * \mathrm{cont}(L) \subseteq c(L)$ for all $L \in \mathrm{CONT}$ (since $L' \in \mathrm{cont}(L)$ implies $L' \subseteq L$), showing that c is right-invariant.

Assertion (3) follows from (2) by Corollary 1.3(1). □

Note that right-invariant control principles need not be extensional as the following proposition shows:

Proposition 5.4. (1) *The control principles* imp, fair, just, pfin, crs *are right-invariant and left-invariant.*

(2) *The control principles* dfr, live, preg, prec, pren, fnl *are right-invariant but not left-invariant.*

(3) *The control principles* rfair, rjust *are left-invariant but not right-invariant.*

(4) *The control principles* nbl *and* aut (*for all classes* **aut** *of control automata*) *are neither right-invariant nor left-invariant.*

The details on the proof (using the definitions and Corollary 5.3) are left to the reader.

It should be remarked that $c * c' \leqslant c$ (and similarly for $c' * c \leqslant c$) does not guarantee to obtain a c-control by the corresponding refinement from a c'-control. We may have $c(L') = \emptyset$ for some or all $L' \in c'(L)$. But if $c(L') \neq \emptyset$ for $L' \in c'(L)$, then we have $L'' \in c(L)$ for all $L'' \in c(L')$, i.e. all controls that we can obtain by the refinement have the desired property.

In general, we make stepwise refinements by $c * c'$ in order to satisfy the c-properties as well as the c'-properties, i.e. we look for $c\&c'$-controls. The following theorem points to such possibilities.

Theorem 5.5. (1) *If c is extensional, then we have $c\&c' \leqslant c * c'$.*

(2) *If c is right-invariant and c' is left-invariant, then we have $c * c' \leqslant c\&c'$.*

(3) *If c is extensional and c' is left-invariant, then we have $c * c' = c\&c'$.*

(4) *If c and c' are both right-invariant (or both left-invariant), then we have $(c * c')\&(c' * c) \leqslant c\&c'$.*

Proof. (1) We have $c\&c'(L) = c(L) \cap c'(L) = \mathrm{cont}(L) \cap C \cap c'(L) = C \cap c'(L) \subseteq C \cap \mathrm{cont}(c'(L)) = c(c'(L)) = c * c'(L)$.

(2) follows by $c * c' \leqslant c$ and $c * c' \leqslant c'$.

(3) follows from (1) and (2).

(4) is proved similar to the proof of (2). □

As applications of Theorem 5.5 we have, for example, live$*$crs = live&crs (but crs$*$live $\not\leqslant$ live!) and crs$*$rfair $\leqslant$ crs&rfair (but rfair$*$crs $\not\leqslant$ rfair!). Hence, the order of stepwise refinements may be important. We furthermore remark that we have imp$*$dfr $\not\leqslant$ dfr and dfr$*$imp $\not\leqslant$ imp although imp is extensional and left-invariant while dfr is right-invariant (even extensional). This shows that the conditions in Theorem 5.5 are necessary.

Since the control principles aut are neither right-invariant nor left-invariant, properties may change (and may be changed) by controlling systems by finite automata.

Another approach to the preservation of properties is due to fixed point considerations. The set C of an extensional control principle c is the greatest fixed point of c (considered as a mapping from $P(\mathrm{CONT})$), and we have the following theorem.

Theorem 5.6. *If c is an extensional control principle, then it holds for arbitrary control principles c':*

$$c' * c \leqslant c \text{ iff } c'(C) \subseteq C.$$

Proof. By $c' * c \leqslant c$ and $C = c(T^*)$ (by Corollary 1.3(2)) we have: $c'(C) = c'(c(T^*)) = c' * c(T^*) \subseteq c(T^*) = C$. On the other hand, by $c'(C) \subseteq C$ we have $c' * c(L) = c'(c(L)) = c'(\mathrm{cont}(L) \cap C) \subseteq C$ and (trivially) $c' * c(L) \subseteq \mathrm{cont}(L)$. Hence, it holds $c' * c(L) \subseteq \mathrm{cont}(L) \cap C = c(L)$. □

Applications like preg$*$live $\not\leqslant$ live are left to the reader.

6. Unitarity

Different elements in $c(L)$ point to different possible c-controls for L. Nevertheless, we may sometimes speak of 'the' c-control for L if there exists a language $L' \in c(L)$ such that all $L'' \in c(L)$ are subsets of L'. Such a maximum element represents the least restrictive control with respect to c.

Definition 6.1. A control principle c is called *unitary* iff

$$\bigcup c(L) \in c(L) \quad \text{for all } L \in \mathrm{CONT} \text{ with } c(L) \neq \emptyset,$$

i.e. iff a maximum element exists in all nonempty sets $c(L)$.

We denote the maximum element by $L_c := \bigcup c(L)$.

Corollary 6.2. *If c is extensional and C is closed under arbitrary unions, then c is unitary.*

Proposition 6.3. (1) dfr, live *and* nbl *are unitary.*

(2) imp, fair, just, pfin, preg, prec, pren, fnl, rfair, rjust *and* crs *are not unitary.*

Proof. (1) DFR and LIVE are closed under arbitrary unions, hence dfr and live are unitary by Corollary 6.2; nbl is unitary since we always have $L \in \mathrm{nbl}(L)$.

(2) Concerning $c \in \{\mathrm{imp}, \mathrm{fair}, \mathrm{just}, \mathrm{rfair}, \mathrm{rjust}\}$ we consider $L := \{a, b\}^*$ where we have: $\bigcup c(L) = L$, since $\overleftarrow{u(ab)^\omega} \in c(L)$ and hence $u \in \bigcup c(L)$ for all $u \in L$ but $L \notin c(L)$ (since $a^\omega \in \mathrm{Adh}(L)$).

Concerning $c \in \{\mathrm{pfin}, \mathrm{preg}, \mathrm{prec}, \mathrm{pren}\}$ we consider a not recursively enumerable infinite sequence w (as in the proof of Theorem 4.4, for example). Then we have $\overleftarrow{\{u\}} \in c(L)$ for all $u \sqsubseteq w$, but $\bigcup c(L) = L \notin c(L)$, where $L := \overleftarrow{\tilde{w}}$.

Finally, for $c \in \{\mathrm{fnl}, \mathrm{crs}\}$, we may consider $L := \overleftarrow{\{b\} \cdot \{a\}^* \cup \{a\}}$, where $\overleftarrow{\{b\} \cdot \{a\}^*}$ and $\{a\}$ are in $c(L)$, but $\bigcup c(L) = L \notin c(L)$. □

Remarks. (1) Since the fairness control principles are not unitary, it makes no sense (in general) to speak of 'the' fair control of concurrent systems as it is sometimes suggested by the 'choice of all fair computations' in order to perform controls.

(2) There is a remarkable difference between crs and the other non-unitary control principles. In $\mathrm{crs}(L)$ we always have maximal elements such that each $L' \in \mathrm{crs}(L)$ is covered by a maximal element. This means that, in order to 'improve' a control by making it less restrictive, we can always find a (relatively) best control in $\mathrm{crs}(L)$. In general, this is not the case for the other non-unitary control principles: there we can improve the controls by making them less restrictive and there need not exist an end of such improvements (or: in the last resort, the resulting control will not have the corresponding properties anymore). As examples we may consider the sequence of finite languages $\{\overleftarrow{a^i}\}$ and the sequence of fair languages $\bigcup_{j \leq i} \overleftarrow{a^j(ba)^\omega}$ for $i \in \mathbb{N}$.

(3) The notion of unitarity was defined regarding all languages from CONT. It might be interesting for which classes of languages the control principles mentioned in Proposition 6.3(2) become unitary (trivial example: preg is unitary with respect to the class PREG, since $\bigcup \mathrm{preg}(L) = L \in \mathrm{preg}(L)$ for all $L \in \mathrm{PREG}$).

Now we are going to study L_{dfr} and L_{live}. We only formulate the results (and the proofs) for L_{live}. The same results hold for L_{dfr} if we omit the statements about the occurrences of $t \in T$ and replace live by dfr.

Theorem 6.4. *We have, for* $L_{(\mathrm{live})} := \{u \mid \exists w \in \mathrm{Adh}(L) : u \sqsubseteq w \wedge \pi_w = \boldsymbol{\omega}\}$,

(1) $L_{(\mathrm{live})} = L_{\mathrm{live}}$ *iff* $\mathrm{live}(L) \neq \emptyset$,

(2) $L_{(\mathrm{live})} = \emptyset$ *iff* $\mathrm{live}(L) = \emptyset$,

(3) $L_{(\mathrm{live})} = \{u \mid \exists u', v \in T^* : uu'v^\omega \in \mathrm{Adh}(L) \wedge \pi_v \geq \mathbf{1}\}$ *iff* $L \in \mathrm{PREG} \cup \mathrm{FNL}$.

Proof. (1), (2): We suppose $\mathrm{live}(L) \neq \emptyset$. If $u \in \bigcup \mathrm{live}(L)$, then we have $u \in L'$ for some $L' \in \mathrm{live}(L)$. By Corollary 1.5(2), there exists an infinite sequence $w \in \mathrm{Adh}(L') \subseteq \mathrm{Adh}(L)$ with $u \sqsubseteq w$ and $\pi_w = \boldsymbol{\omega}$ such that $u \in L_{(\mathrm{live})}$. Thus we have $\bigcup \mathrm{live}(L) \subseteq L_{(\mathrm{live})}$, by the definition of $L_{(\mathrm{live})}$ and hence (by Corollary 1.5(2)) $L_{(\mathrm{live})} \in \mathrm{LIVE}$.

Furthermore, $L_{(\text{live})}$ is contained in L and therefore $L_{(\text{live})} \in \text{live}(L)$, i.e. $L_{(\text{live})} \subseteq \bigcup \text{live}(L) = L_{\text{live}}$.

If we suppose $u \in L_{(\text{live})} \neq \emptyset$, then we have $w \in \text{Adh}(L)$ with $\pi_w = \omega$ such that $\tilde{w} \in \text{live}(L) \neq \emptyset$.

(3) If $L \in \text{PREG} \cup \text{FNL}$, then we have

$$\exists w \in \text{Adh}(L): u \sqsubseteq w \wedge \pi_w = \omega \text{ iff } \exists u', v \in T^*: uu'v^\omega \in \text{Adh}(L) \wedge \pi_v \geq \mathbf{1}$$

(by the same proof as for Proposition 1.6(2)). □

Theorem 6.5. *It holds for* $\text{live}(L) \neq \emptyset$:

(1) *If* $L \in \text{PREG}$, *then* $L_{\text{live}} \in \text{PREG}$.

(2) *If* $L \in \text{FNL}$, *then* $L_{\text{live}} \in \text{PREC}$ (*but in general* $\notin \text{FNL}$).

(3) $L \in \text{PREC}$ *does not imply* $L_{\text{live}} \in \text{PREC}$.

Proof. (1) For $L = L_S$ where $S = (T, Q, f, q_0)$ is a finite transition system we put

$$Q' := \{q \mid \exists q' \in Q\ \exists u, v \in T^*: q' \in f(q, u) \wedge q' \in f(q', v) \wedge \pi_v \geq \mathbf{1}\},$$

$$f' := f|_{Q' \times T},$$

$$S' := (T, Q', f', q_0) \text{ (note that } q_0 \in Q' \text{ since } \text{live}(L) \neq \emptyset).$$

Then it can be proved that $L_{\text{live}} = L_{S'}$ hence $L_{\text{live}} \in \text{PREG}$. Note that Q' (and hence S') can be constructed since it suffices to regard u', v with length not greater than $\text{card}(Q)$.

(2) By Theorem 6.4 we have

$$u \in L_{\text{live}} \text{ iff } \exists u' \in T^*\ \exists v \in L_0: uu'v^\omega \in \text{Adh}(L),$$

where $L_0 := \{v \mid \pi_v \geq \mathbf{1}\}$ is a regular language. Thus, we can apply Lemma 2.3 and show that L_{live} is recursive. Theorem 6.7 will show that L_{live} need not be in FNL.

(3) We consider a recursive set $M \subseteq \mathbb{N} \times \mathbb{N}$ with the property that the set $\{n \mid \forall i \in \mathbb{N}: (n, i) \in M\}$ is not recursively enumerable. Then we have:

$$L := \overleftarrow{\{a^n(ba)^m \mid n \in \mathbb{N} \wedge \forall i \leq m: (n, i) \in M\}} \in \text{PREC},$$

$$L_{\text{live}} = \overleftarrow{\{a^n(ba)^m \mid m, n \in \mathbb{N} \wedge \forall i \in \mathbb{N}: (n, i) \in M\}} \notin \text{PREN}. \quad \square$$

Corollary 6.6. *The problem* "$\text{live}(L) \neq \emptyset$" *is decidable for* $L \in \text{PREG} \cup \text{FNL}$.

Proof. By Theorem 6.4 we have $\text{live}(L) \neq \emptyset$ iff $L_{(\text{live})} \neq \emptyset$, i.e. iff $e \in L_{(\text{live})}$. Referring to the proof of Theorem 6.5 we have $e \in L_{(\text{live})}$ iff $q_0 \in Q'$ in the case of $L \in \text{PREG}$ and $e \in L_{(\text{live})}$ iff $\exists u' \in T^*\ \exists v \in L_0: u'v^\omega \in \text{Adh}(L)$ in the case $L \in \text{FNL}$. We remark that this result for FNL has been proved also in [20, Theorem 3.12] and in [9, Theorem 2]. □

Theorem 6.7. (1) *There exists a language* $L \in \mathrm{FNL}$ *for* $\mathrm{card}(T) \geq 3$ *with* $\mathrm{live}(L) \neq \emptyset$ *and* $\mathrm{live}\&\mathrm{fnl} = \emptyset$.

(2) *For* $L \in \mathrm{PREG} \cup \mathrm{FNL}$ *with* $\mathrm{live}(L) \neq \emptyset$ *we have* $\mathrm{live}\&\mathrm{preg}(L) \neq \emptyset$.

Proof. (1) For the Petri net N of Fig. 4 we have $L_{\mathrm{live}} = \overrightarrow{\{abbaac\}^*} \notin \mathrm{FNL}$ (cf. the proof of $\mathrm{FNL} \neq \mathrm{aut1(FNL)}$ of Theorem 4.4) and $\mathrm{live}(L) = \{L_{\mathrm{live}}\}$ $(= \mathrm{dfr}(L))$.

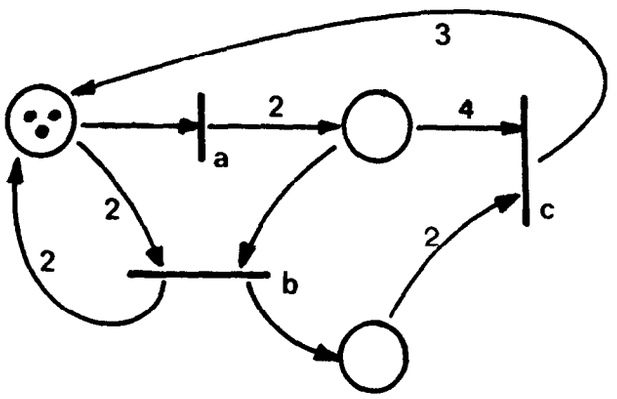

Fig. 4.

(2) is a consequence of Theorem 6.4(3): For $u = e \in L_{\mathrm{live}}$ we have u', v with $u'v^\omega \in \mathrm{Adh}(L)$ and $\pi_v \geq \mathbf{1}$, hence $\overrightarrow{\{u'\} \cdot \{v\}^*} \in \mathrm{live}(L)$. □

In conclusion of Theorem 6.7, the Petri net model may be not sufficient to model live controlled Petri nets as long as we do not use additional transitions as in [20]. But there are possibilities to model Petri nets under certain live controls by finite transition systems or to model certain live controls by finite control automata (cf. Theorem 7.3(2) below).

Remark. The non-unitarity of the fairness control principles can be regarded as the consequence of the freedom to have arbitrarily long delays. Following some ideas in [14] we can consider delay functions $d : T^* \times T \to \mathbb{N}$ and define impartiality, fairness and justice with respect to a given delay function d by

$$L \in d\text{-IMP} \quad \text{iff} \quad \forall t \in T\, \forall uv \in L\colon |v| > d(u, t) \to \pi_v(t) > 0,$$

$$L \in d\text{-FAIR} \quad \text{iff} \quad \forall t \in T\, \forall uv \in L\colon$$

$$\mathrm{card}(\{v' \mid v' \sqsubsetneq v \wedge uv't \in L\}) > d(u, t) \to \pi_v(t) > 0,$$

$$L \in d\text{-JUST} \quad \text{iff} \quad \forall t \in T\, \forall uv \in L\colon$$

$$(|v| > d(u, t) \wedge \forall v' \sqsubsetneq v\colon uv't \in L) \to \pi_v(t) > 0.$$

Then the extensional control principles d-imp, d-fair, d-just are unitary.

We have $\bigcup_d d\text{-IMP} = \mathrm{IMP}$ and $\bigcup_d d\text{-JUST} = \mathrm{JUST}$, but only $\bigcup_d d\text{-FAIR} \subsetneq \mathrm{FAIR}$ (cf. [9]), i.e. fairness is not completely expressible by delay functions in the given way. Similar results hold for the analogously definable non-extensional control principles d-rfair and d-rjust.

7. Existence of controls

By the problem "$c(L) \neq \emptyset$?" we ask for the existence of controls (specified by c) for given systems (specified by L).

Theorem 7.1. *It holds for arbitrary languages* $L \in \mathrm{CONT}$:

(1) $c(L) \neq \emptyset$ *for* $c \in \{\mathrm{imp}, \mathrm{fair}, \mathrm{just}, \mathrm{rfair}, \mathrm{rjust}\}$.

(2) $c\&\mathrm{nbl}(L) \neq \emptyset$ *for* $c \in \{\mathrm{fair}, \mathrm{just}, \mathrm{rfair}, \mathrm{rjust}\}$.

(3) $c\&\mathrm{dfr}(L) \neq \emptyset$ *iff* $\mathrm{dfr}(L) \neq \emptyset$ *for* $c \in \{\mathrm{fair}, \mathrm{just}\}$.

(4) $\mathrm{imp}\&\mathrm{dfr}(L) \neq \emptyset$ *iff* $\mathrm{live}(L) \neq \emptyset$.

(5) $\mathrm{imp}\&\mathrm{nbl}(L) \neq \emptyset$ *iff* $\mathrm{imp}\&\mathrm{dfr}(L) \neq \emptyset \vee \mathrm{pfin}\&\mathrm{nbl}(L) \neq \emptyset$.

Proof. (1) holds by $\mathrm{pfin} \leqslant c$ for the fairness control principles. (2) is shown by the application of appropriate queue regimes. Such queues can be understood as sequences $u \in T^*$ with $\pi_u \leqslant \mathbf{1}$, for example. The actions t appearing in u (i.e. $\pi_u(t) = 1$) are the waiting actions. If we have the actual queue u and the set U of actually performable actions (in the uncontrolled system), then the work is defined as follows.

We build a sequence v from those actions which belong to U and are not in the queue u. Then the first action t from uv with $t \in U$ is performed next and the queue u' for the following step is built by deleting t from uv.

Different queue regimes are founded on different possibilities to build the sequences v. If they are built in some regular way, the queue regimes can be performed by finite control automata (with the queues as states).

Obviously, we obtain (relatively) fair and non-blocking controls by the queue regimes as described above. If all those actions not appearing in U (they are actually not performable in the uncontrolled system) are additionally deleted from the queue u' during the reorganization step, then we get a (relatively) justice and non-blocking control. (Petri nets under related firing rules have been studied in [5].) We mention that the results of such controls for a language L are in d-fair(L) and d-just(L), respectively, where $d(u, t) = \mathrm{card}(T) - 1$ for $u \in T^*$, $t \in T$.

(3) If $\mathrm{dfr}(L) \neq \emptyset$, then the application of the queue regimes defined in the proof, part (2), to a language $L' \in \mathrm{dfr}(L)$ results in a language $L'' \in c * \mathrm{dfr}(L)$. We have $L'' \in \mathrm{dfr}(L)$ since the queue regimes work non-blocking, and we have $L'' \in c(L)$, i.e. $L'' \in c\&\mathrm{dfr}(L)$, since fair and just are right-invariant (by Proposition 5.4). We remark that $\mathrm{dfr}(L) \neq \emptyset$ does not imply $\mathrm{dfr}\&\mathrm{rfair}(L) \neq \emptyset$ and $\mathrm{dfr}\&\mathrm{rjust}(L) \neq \emptyset$, respectively (example: $L := \overleftarrow{\{a\}^* \cdot \{b\}}$).

(4) If $\mathrm{imp}\&\mathrm{dfr}(L) \neq \emptyset$, then $\mathrm{live}(L) \neq \emptyset$ by Theorem 3.4. If $\mathrm{live}(L) \neq \emptyset$, then there exists a $w \in \mathrm{Adh}(L)$ with $\pi_w = \boldsymbol{\omega}$ (by Theorem 6.4, i.e. $\tilde{w} \in \mathrm{imp}\&\mathrm{dfr}(L)$.

(5) We suppose $L' \in \mathrm{imp}\&\mathrm{nbl}(L) \neq \emptyset$. If $\mathrm{imp}\&\mathrm{dfr}(L) = \emptyset$, then $L' \notin \mathrm{dfr}(L)$, i.e. there exists an $u \in L'$ such that $ut \notin L'$ for all $t \in T$. It follows by $L' \in \mathrm{nbl}(L)$ that even $ut \notin L$ for all $t \in T$, and hence we have $\overleftarrow{\{u\}} \in \mathrm{pfin}\&\mathrm{nbl}(L) \neq \emptyset$.

The reverse direction follows by

$$\text{imp\&nbl}(L) \supseteq \text{imp\&dfr}(L) \cup \text{pfin\&nbl}(L). \qquad \square$$

For those of the control principles c mentioned in Theorem 7.1 where $c(L)$ may be empty (corresponding controls need not exist for each system), we can decide the existence of controls in the case of finite transition systems and Petri nets.

Theorem 7.2. *The problems* "$c(L) \neq \emptyset$?" *are decidable for* $L \in \text{PREG} \cup \text{FNL}$ *and* $c \in \{\text{dfr, live, imp\&dfr, fair\&dfr, just\&dfr, pfin\&nbl, imp\&nbl}\}$.

Proof. The result for live was given by Corollary 6.6 and as mentioned before Theorem 6.4, the result holds for dfr, too. The results for imp&dfr, fair&dfr, just&dfr follow then immediately by Theorem 7.1(3)(4).

We have $\text{pfin\&nbl}(L) \neq \emptyset$ iff $L \notin \text{DFR}$ by Corollary 3.6(3), hence the problem is decidable for the case pfin&nbl by Theorem 2.1.

Finally, the result holds for imp&nbl as a consequence of Theorem 7.1(5). $\square$

Remark. The problems "$\text{rfair\&dfr}(L) \neq \emptyset$?" and "$\text{rjust\&dfr}(L) \neq \emptyset$?" are also decidable for languages $L \in \text{PREG}$ (by using methods as in the proof of Theorem 2.1), the problems are open for languages $L \in \text{FNL}$.

Theorem 7.3. (1) *A control automaton* $\boldsymbol{A}$ *can be constructed for each control principle* $c \in \{\text{fair, just, rfair, rjust}\}$ *such that* $L/\boldsymbol{A} \in c\&\text{nbl}(L)$ *holds for all* $L \in \text{CONT}$.

(2) *A control automaton* $\boldsymbol{A}$ *can be constructed for each* $c \in \{\text{dfr, live, pfin\&nbl, imp\&nbl, imp\&dfr, fair\&dfr, just\&dfr}\}$ *and each language* $L \in \text{PREG} \cup \text{FNL}$ *such that* $L/\boldsymbol{A} \in c(L)$ *holds if* $c(L) \neq \emptyset$.

Proof. (1) As mentioned in the proof for Theorem 7.1(2), certain queue regimes can be realized by control automata $\boldsymbol{A}$ such that each system can be controlled by $\boldsymbol{A}$ in the sense of (relative) fairness and (relative) justice, respectively.

(2) If $\text{live}(L) \neq \emptyset$ (and similarly if $\text{dfr}(L) \neq \emptyset$), then there exists some $L' \in \text{live\&preg}(L)$ by Theorem 6.7(2).

Following the proofs up to Lemma 2.3 we find that such languages L' can be constructed in the form $L' = \overleftarrow{uv^\omega}$ with $L' \in \text{preg\&nbl}(L)$. A corresponding control automaton $\boldsymbol{A}$ with $L/\boldsymbol{A} = L'$ can then be constructed according to Proposition 4.6 (note that the proof of Proposition 4.6 is constructive). If $\text{pfin\&nbl}(L) \neq \emptyset$, then a sequence u with $\{u\} \in \text{pfin\&nbl}(L)$ can be found (trivially by successive tests) and then we can apply Proposition 4.6 again.

If $\text{imp\&nbl}(L) \neq \emptyset$, then we have $\emptyset \neq \text{imp\&dfr}(L) \cup \text{pfin\&nbl}(L) \subseteq \text{imp\&nbl}(L)$ (cf. Theorem 7.1(5), and we can decide whether $\text{imp\&dfr}(L) \neq \emptyset$ or $\text{pfin\&nbl}(L) \neq \emptyset$ holds (by Theorem 7.2); then we can proceed as for imp&dfr and for pfin&nbl, respectively.

If imp&dfr(L) $\neq \emptyset$, we have live(L) $\neq \emptyset$ by Theorem 7.1(4) and (as shown above) we can find $L' = \overleftarrow{uv^\omega} \in$ live(L). Obviously, it holds that $L' \in$ imp&dfr(L) and we can proceed as for live. Finally, if fair&dfr(L) $\neq \emptyset$ and just&dfr(L) $\neq \emptyset$, respectively, then we have dfr(L) $\neq \emptyset$ and we can find some $L' \in$ dfr(L) where L' is of the form $L' = \overleftarrow{uv^\omega}$ such that Proposition 4.6 can be applied (all proofs similar to the proof for live). Since $\tilde{w} \in$ fair&dfr(L) ($\in$ just&dfr(L)) for all $w \in$ Adh(L), the proof is finished. □

Remark. Theorem 7.3(2) also holds for $c \in$ {rfair&dfr, rjust&dfr} in the case $L \in$ PREG. It does not hold in the case $L \in$ FNL [11].

The fact that the control automata with respect to the control princples of Theorem 7.3(2) must be constructed individually for the languages $L \in$ PREG $\cup$ FNL (while we have uniform control automata in the case of Theorem 7.3(1)) corresponds to the result in [12] that there exists no conflict resolving regular firing strategy for Petri nets which supports deadlock avoidance and liveness, respectively.

Remark. There exist Petri nets N such that the maximum element in live(L_N) cannot be realized under the control of our control automata (i.e. for $L = L_N$ we have live(L) $\neq \emptyset$, but there exists no control automaton A with $L/A = L_{\text{live}}$; the same holds for dfr). An example is given by Fig. 5.

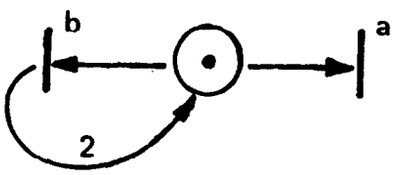

Fig. 5.

Since the input information of a control automaton A is always $U = \{a, b\}$ or $U = \emptyset$, no such control automaton can produce

$$L/A = L_{\text{live}} = \{u \mid u \in \{a, b\}^* \wedge \forall v \sqsubseteq u: \pi_v(a) > \pi_v(b) \geq 0\}.$$

It is interesting to note that the situation changes if we use automata recognizing differences concerning concurrent firability (by inputs $\{\{a, b\}\}$ for $m(p) \geq 2$ and $\{\{a\}, \{b\}\}$ for $m(p) = 1$ as in [7] for example): Such automata can realize a control resulting in L_{live} for the net of Fig. 5 (but again, they cannot do so if we add a run place to the net such that the transitions a and b are never concurrently firable).

While we are able to construct automata controls for the control principles referred to in Theorem 7.3 in the case of Petri nets (with possibly great effort), we are not able to decide whether a given control automaton for a Petri net realizes a corresponding control. This can be understood in that sense that the construction of an appropriate control may be easier than the verification of a property for a given control. We prove this by the following theorem.

Theorem 7.4. *We suppose that* card(T) *is sufficiently large. (The exact lower bound is unknown. The method of proof gives lower bounds between* $2n$ *and* $6n+47$ *depending on the control principles, where* n *is the number of states of a deterministic counter machine computing a function with a non-recursive domain.)*

(1) *Let* A *be a control automaton such that* $L/A \in \mathrm{crs}(L)$ *for all* $L \in \mathrm{FNL}$ *(i.e.* A *represents a conflict resolution rule for Petri nets). Then, for each control principle* $c \in \{\mathrm{dfr}, \mathrm{live}, \mathrm{imp}, \mathrm{imp\&dfr}, \mathrm{imp\&nbl}, \mathrm{pfin}, \mathrm{pfin\&nbl}\}$, *the problems "*$L/A \in c(L)$?*" are undecidable for* $L \in \mathrm{FNL}$.

(2) *For each control principle* $c \in \{\mathrm{dfr}, \mathrm{live}, \mathrm{imp}, \mathrm{fair}, \mathrm{just}, \mathrm{rfair}, \mathrm{rjust}, \mathrm{imp\&nbl}, \ldots, \mathrm{rjust\&nbl}, \mathrm{imp\&dfr}, \ldots, \mathrm{rjust\&dfr}, \mathrm{pfin}, \mathrm{pfin\&nbl}, \mathrm{preg}, \mathrm{crs}\}$ *the problems "*$L/A \in c(L)$?*" are undecidable for* $L \in \mathrm{FNL}$ *and arbitrary control automata* A.

Proof. The proof uses the simulation of deterministic counter machines by automata controlled Petri nets. For more details of such simulations the reader is referred to [7, 8, 11]. Throughout the proof, all results for a control principle c are results for c&nbl by Proposition 4.2.

(1) There are counter machines M with two counters such that the Halting problem is undecidable with respect to different initial counter contents. If M has n states and $A=(P(T), T, Z, h, z_0)$ is a control automaton whereby card(T) = $3n+28$, then a Petri net N can be constructed such that M is simulated in some sense by N under the control of A if $L_N/A \in \mathrm{crs}(L_N)$. Different initial counter contents of M can be simulated by different initial markings in N. By the simulation we obtain $L_N/A \in \mathrm{pfin}(L_N)$ iff the initial marking of N corresponds to initial counter contents of M for which M stops. We obtain $L_N/A \in \mathrm{dfr}(L_N)$ iff M does not stop for the corresponding initial counter contents. The undecidability of the Halting problem implies the undecidability of '$L_N/A \in c(L_N)$' for $c = \mathrm{dfr}, \mathrm{pfin}, \mathrm{pfin\&nbl}$.

The result concerning liveness was proved in [12] by construction of an appropriate net N' from N, thereby we need card(T) $\geq 6n+47$. (The net N' is constructed in such a way that $L_{N'}/A \in \mathrm{live}(L_{N'})$ iff the underlying counter machine does not stop for the corresponding intial counter contents, whereby only the assumption $L_{N'}/A \in \mathrm{crs}(L_{N'})$ is made concerning A.)

The result for imp&dfr follows by Theorem 7.1(4).

The net N constructed in [7] for the simulation of M has a 'stop-place' p_{stop} such that the net N becomes dead during the simulation run iff the place p_{stop} is marked indicating that M stops. Now, we add a transition t to N which takes a token from p_{stop} and then we have for the new net N'': $L_{N''}/A$ is impartial (and also nonblocking with respect to $L_{N''}$) if M stops (by pfin $\leq$ imp), but it is not impartial if M does not stop (by neglecting the new transition t).

(2) Since (1) is a stronger result, it remains to prove (2) for $c = \mathrm{fair}, \mathrm{just}, \mathrm{rfair}, \mathrm{rjust}$ (implying the result for fair&nbl, ..., rjust&nbl), fair&dfr, ..., rjust&dfr, preg, and crs. We can use the simulation of deterministic counter machines by a priority firing rule (cf. [16]) which can easily be realized by control automata having only one state. The simulating net does not need more than $2n$ transitions. Again, we

can construct to a deterministic counter machine M (for which the Halting problem is not decidable) and a control automaton $A=(P(T), T, \{z_0\}, h, z_0)$ realizing a deterministic priority firing rule over T with $\text{card}(T)=2n$) a Petri net N with a place p_{stop} that can be marked by one token iff M stops. The net N becomes dead during a simulation run iff p_{stop} is marked. The proof can be finished by appropriate simple supplements of N and A as given in [11]. □

Remark. The stronger result in Theorem 7.4(1) cannot hold for $c=$ fair, ..., rjust, fair&nbl, ..., rjust&nbl (otherwise we would contradict Theorem 7.3(1)). By related arguments, Theorem 7.3(1) cannot hold for the control principles referred to in Theorem 7.4(1) if we consider conflict resolving automata. Moreover, the result as in Theorem 7.4(1) cannot hold for crs (since, for example, all deterministic control automata work conflict resolving). The problem whether an assertion as in Theorem 7.4(1) holds is open for $c=$ preg, fair&dfr, ..., rjust&dfr.

Finally, the properties which are undecidable for automata controlled Petri nets can be proved to be decidable in the case of finite transition systems (and thus, for bounded Petri nets, too).

Theorem 7.5. *Let c be a control principle as in Theorem* 7.4(2). *The problem "*$L/A \in c(L)$*?" is decidable for arbitrary languages* $L \in$ PREG *and arbitrary control automata* A.

Proof. We have $L/A \in$ PREG for all $L \in$ PREG and all control automata A by Theorem 4.4. Hence, Theorem 7.5 is a consequence of Theorem 3.7 and the related results for $c \in$ {rfair, rjust, crs, nbl} proved in [11]. □

8. Conclusions

It was shown that many problems of control can be studied on a very high level of abstraction. This framework should be useful to study and to compare different approaches to systems, controls and their properties, as for example given in [1, 3, 5, 12, 13, 14, 18, 19].

The use of our approach is restricted by considering systems to be equivalent if they have the same external behaviour regarding abstract languages (but there may be possibilities for different controls based on different internal structures). In some cases this problem can be overcome by other notions of external behaviour. To be closer to concurrency, the alphabet T may be chosen as the powerset of another set of actions. In such models the max-semantics [19, 5] can be considered (with appropriate definitions reflecting liveness and fairness properties). Another approach to concurrency by partially ordered sequences may lead to the study of the corresponding behaviour and controls with regard to languages of partially ordered sequences. The general notion of control principles opens the way to further

examinations of properties like invariance and unitarity; stochastic controls may be studied by measuring the sets $c(L)$.

Acknowledgment

I want to express my thanks to P.H. Starke and H.W. Pohl for various helpful discussions.

References

[1] A. Arnold and M. Nivat, Controlling behaviours of systems, some basic concepts and some applications, in: P. Dembiński, ed., *Mathematical Foundations of Computer Science*, Lecture Notes in Comp. Sci. **88** (Springer, Berlin, 1980) 113–122.

[2] K.R. Apt, A. Pnueli and J. Stavi, Fair termination revisited—with delay, *Publ. du L.I.T.P.* (Univ. Paris VII, 1982) 82–51.

[3] E. Best, Relational semantics of concurrent programs (with some applications), *Proc. IFIP TC-2 Conf. on Formal Descriptions of Programming Concepts*, Garmisch-Partenkirchen, 1982.

[4] L. Budach, Environments, labyrinths and automata, in: M. Karpiński, ed., *Fundamentals of Computation Theory*, Lecture Notes in Comp. Sci. **56** (Springer, Berlin, 1977) 54–64.

[5] H.D. Burkhard, Ordered firing in Petri nets, *Elektron. Informationsverarbeitung und Kybernetik* **17**(2/3) (1981) 71–86.

[6] H.D. Burkhard, Two pumping lemmata for Petri nets, *Elektron. Informationsverarbeitung und Kybernetik* **17**(7) (1981) 349–362.

[7] H.D. Burkhard, What gives Petri nets more computational power, Preprint 45, Sekt. Mathematik d. Humboldt-Univ. Berlin, 1982.

[8] H.D. Burkhard, Control of Petri nets by finite automata, *Fundamenta Informaticae* **VI.2** (1983) 185–215.

[9] H.D. Burkhard, On the control of concurrent systems with respect to fairness and liveness conditions, *Proc. Internat. Summer School of the Programming Language LOGLAN-82*, Zaborów, Poland, 1983.

[10] H.D. Burkhard, Fair and live controls for finite transition systems, *Internat. Symp. on 'Diskrete Mathematik'*, Berlin, 1983, in: *Seminarberichte der Sekt. Math.* **56** (Humboldt-Univ. Berlin, 1984) 9–16.

[11] H.D. Burkhard, Untersuchung von Steuerungsproblemen nebenläufiger Systeme auf der Basis abstrakter Steuersprachen, in: *Seminarberichte der Sekt. Math.* **58** (Humboldt-Univ. Berlin, 1984).

[12] H.D. Burkhard and P.H. Starke, A note on the impact of conflict resolution to liveness and deadlock in Petri nets, *Fundamenta Informaticae* **VII.4** (1984).

[13] L. Czaja, Are infinite behaviours of parallel system schemata necessary?, in: A. Salwicki, ed., *Logics of Programs and Their Application*, Lecture Notes in Comp. Sci. **148** (Springer, Berlin, 1983) 108–117.

[14] H. Carstensen and R. Valk, Infinite behaviour and fairness in Petri nets, in: *Proc. 4th European Workshop on Application and Theory of Petri Nets*, Toulouse (1983) 104–123.

[15] M. Hack, The recursive equivalence of the reachability problem and the liveness problem for Petri nets and vector addition systems, in: *15th Ann. Symp. on Switching and Automata Theory* (1974) 156–164.

[16] M. Hack, Petri net languages, C.S.G. Memo 124, Project MAC, M.I.T., 1975.

[17] S.R. Kosaraju, Decidability of reachability in vector addition systems, in: *Proc. 14th Ann. ACM Symp. on Theory of Computing* (1982) 267–281.

[18] D. Lehmann, A. Pnueli and J. Stavi, Impartiality, justice, fairness: The ethics of concurrent termination, in: S. Even and O. Kariv, eds., *Automata, Languages and Programming*, Lecture Notes in Comp. Sci. **115** (Springer, Berlin, 1981) 264–277.

[19] A. Salwicki and T. Mueldner, On the algorithmic properties of concurrent programs, in: E. Engeler, ed., *Logic of Programs*, Lecture Notes in Comp. Sci. **125** (Springer, Berlin, 1981) 169–197.

[20] R. Valk and M. Jantzen, The residue of vector sets with applications to decidability problems in Petri nets, Bericht IFI-HH-B-101/84, Fachbereich Informatik, Univ. Hamburg (earlier version in: *Proc. 4th Europ. Workshop on Applikation and Theory of Petri Nets*, Toulouse (1983) 342-363).

[21] R. Valk and G. Vidal-Maquet, Petri nets and regular languages, *J. Comput. System. Sci.* **23**(3) (1981) 299-325.