PRACE IPI PAN • ICS PAS REPORTS

Józef Winkowski

TOWARDS A SOLUTION TO A PLAN-FORMATION PROBLEM*

Nr 948

Warsaw, September 2002

INSTYTUT PODSTAW INFORMATYKI
POLSKIEJ AKADEMII NAUK

INSTITUTE OF COMPUTER SCIENCE
POLISH ACADEMY OF SCIENCES

ul. Ordona 21
01-237 Warszawa
tel.: ++(48-22) 836-28-41
http://www.ipipan.waw.pl

Józef Winkowski

TOWARDS A SOLUTION TO A PLAN-FORMATION PROBLEM*

Nr 948

Warsaw, September 2002

Abstract: The paper deals with the problem of achieving a goal in a system in which certain actions can be executed. The initial state and the states that may be reached from this state by executing actions are represented by sets of sentences, each sentence saying that some objects are or are not in a relation. Available actions are represented as instances of rewriting rules. The problem consists in finding a sequence of actions that leads from the initial state to a state in which the goal is achieved. The presented solution to this problem for a class of systems is based on reducing it to the problem of reachability in contextual Petri nets.

Keywords: system, state, action, goal, literal, situation, rewriting rule, relational rewriting system, safe rewriting system, contextual net, safe contextual net, net morphism, contextual occurrence net, process, reachability.

ZARYS PEWNEJ METODY PLANOWANIA AKCJI

Streszczenie: Praca dotyczy problemu osiągania celu w systemie, w którym można wykonywać pewne akcje. Stan początkowy i stany, które mogą być z tego stanu osiągnięte przez wykonywanie akcji, są reprezentowane przez zbiory zdań, z których każde orzeka, że pewne obiekty są lub nie są w pewnych relacjach. Możliwe akcje są reprezentowane jako instancje pewnych reguł przekształcania. Problem polega na znalezieniu ciągu akcji prowadzącego ze stanu początkowego do stanu, w którym cel jest osiągnięty. Proponowane rozwiązanie tego problemu dla pewnej klasy systemów polega na jego redukowaniu do problemu osiągalności w kontekstowych sieciach Petriego.

Słowa kluczowe: system, stan, akcja, cel, literał, sytuacja, reguła przekształcania, relacyjny system przekształcania, bezpieczny system przekształcania, sieć kontekstowa, bezpieczna sieć kontekstowa, morfizm sieci, kontekstowa sieć wystąpień, proces, osiągalność.

---

*This work has been supported by the Institute of Computer Science of the Polish Academy of Sciences.

The paper is available under the address http://www.ipipan.waw.pl/~wink/ winkowski.htm

1 Introduction

In this paper we present a plan-formation problem as formulated in [Kowa 76] and a way of solving it. More precisely, we deal with the problem of achieving a goal in a system in which certain actions can be executed, that is with the problem of finding a sequence of actions that leads from a given initial state to a state in which the goal is achieved.

The problem is crucial for any approach to planning since its solution is a natural component of any algorithm in this domain. With the aid of such a solution it is possible to try to achieve a goal in conditions in which actions may fail, to find rules according to which actions should be chosen for execution depending on local situations etc. (cf. [Drum 89] and [GoKa 91], for example).

We consider systems whose states can be represented by sets of sentences, each sentence saying that some objects are or are not in some relations, and whose actions can be represented as instances of some rewriting rules similar to those in graph grammars (cf. [EPS 73]). In order to avoid the state explosion that results from describing concurrency of actions as indeterministic interleaving we represent such systems by Petri nets. More precisely, we use for this purpose nets of a special kind, called contextual nets (cf. [MR 95], for the concept). The idea is similar to that in [Drum 89], but more general because the notion of context can be taken into account in an adequate way and, consequently, the independence of actions can be defined in a more subtle way than in standard Petri nets.

1.1. Example. Imagine that two robots A and B are supposed to transport two objects P and Q through a corridor C and to paint the floor of the corridor. Assume that the robots may transport objects provided that the corridor is dry, that is not painted yet, and that they may do it concurrently. States of this system can be represented in terms of constants A, B, P, Q, C and relations *free*, *transported*, *at*, *dry*, *painted*, where $free(x)$, $transported(y)$, $dry(z)$, $painted(z)$, and $at(y,z)$, stand respectively for "robot x is free", "object y is transported", "corridor z is not painted yet", "corridor z is painted", and "object y is to be transported through z". The available actions can be represented as instances of the following rewriting rules for x, y, z ranging over the sets $\{A,B\}$, $\{P,Q\}$, $\{C\}$, respectively:

"replace $\{free(x), at(y,z)\}$ by $\{free(x), transported(y)\}$ in the context $\{dry(z)\}$"

"replace $\{free(x), dry(z)\}$ by $\{free(x), painted(z)\}$".

The initial state and the goal are represented respectively as follows:

$S = \{at(P,C), at(Q,C), dry(C), free(A), free(B)\}$

$S' = \{transported(P), transported(Q), painted(C), free(A), free(B)\}$.

This system can be represented by the contextual net in figure 1. The places of this net (circles) represent possible facts about relations among system components. The transitions (boxes) represent possible actions, that is, in this case, the actions

X = "replace $\{free(A), at(P,C)\}$ by $\{free(A), transported(P)\}$ in the context $\{dry(C)\}$"

Y = "replace $\{free(B), at(Q,C)\}$ by $\{free(B), transported(Q)\}$ in the context $\{dry(C)\}$"

Z = "replace $\{free(A), at(Q,C)\}$ by $\{free(A), transported(Q)\}$ in the context $\{dry(C)\}$"

T = "replace $\{free(B), at(P,C)\}$ by $\{free(B), transported(P)\}$ in the context $\{dry(C)\}$"

U = "replace $\{free(A), dry(C)\}$ by $\{free(A), painted(C)\}$"

V = "replace $\{free(B), dry(C)\}$ by $\{free(B), painted(C)\}$".

The flow relation (directed edges from circles to boxes and from boxes to circles) represents how facts cause actions and how actions cause facts. The context relation (dotted lines between circles and boxes) represents how facts play the role of a context in which the corresponding actions are executable.

The problem of achieving the required goal reduces to finding a sequence of actions from the state S to the state S'. It can be regarded as the problem of reaching the marking representing S' in the contextual net of the system. □

In [Wink 02] it has been shown that the behaviours of systems described in the form of contextual nets can be represented by branching processes similar to those considered in [Eng 91] for standard Petri nets.

1.2. Example. A concrete partial behaviour of the system decribed in 1.1 can be represented by the corresponding branching process of its net, shown as the labelled occurrence net in figure 2. This occurrence net has the set $P = \{1,2,3,4,5,6,7,8,9,10,11,12,13,14,15\}$ of places, the set $T = \{a,b,c,d,e\}$ of transitions, the flow relation $F = \{(1,a),(2,a),(a,3),(a,4),(10,b),(11,b),(b,12),(b,13),(4,c),(10,c),(c,6),(c,7),(5,d),(7,d),(d,8),(d,9),(5,e),(12,e),(e,14),(e,15)\}$, and the context relation $C = \{(5,a),(5,b),(5,c)\}$. The label of each place and transition is written after its identity, separated by a colon. Each place represents a concrete occurrence of the fact specified by the corresponding label. Each transition represents a concrete execution of the corresponding action.

The abstract process corresponding to the concrete process in figure 2, that is the corresponding isomorphism class of concrete processes, is shown in figure 3. It is obtained from the concrete process by forgetting the identities of places and transitions of the underlying occurrence net. The corresponding labels are kept because isomorphisms of labelled nets must preserve labels. □

In the set of abstract branching processes of a contextual net there is a natural prefix relation with respect to which this set is a complete lattice. In particular, there exists the greatest branching process of the net, called the unfolding of the net.

Once the system and its behaviour are represented by a contextual net and its unfolding, the problem of achieving in the system the required goal reduces to the problem of reaching in the contextual net of the system the marking representing the goal.

In [Wink 02] it has been shown that for a finite safe contextual net such a problem can be solved by finding in a way similar to that described in [McM 93] a finite complete prefix of the unfolding of the net, and by checking in a way similar to that in [Espa 93] each branch of such a prefix for the existence of an appropriate marking, that is for the existence of a configuration whose characteristic function is a solution of the corresponding linear programming problem.

In the present paper we formally define systems for which we consider the plan-formation problem, we show how such systems can be transformed into contextual nets, and we show how the plan-formation problem can be solved using the methods developed for contextual nets.

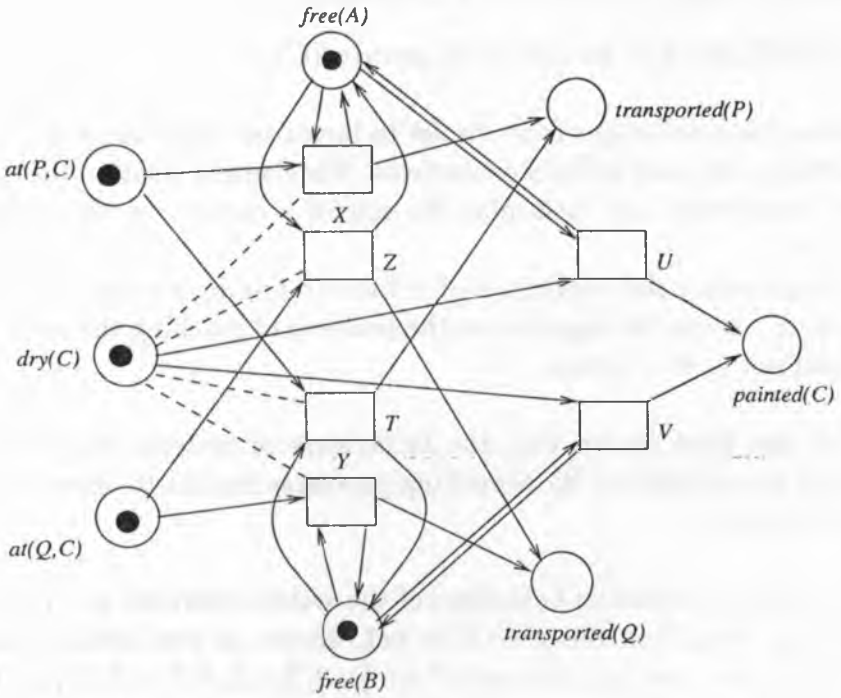

Figure 1

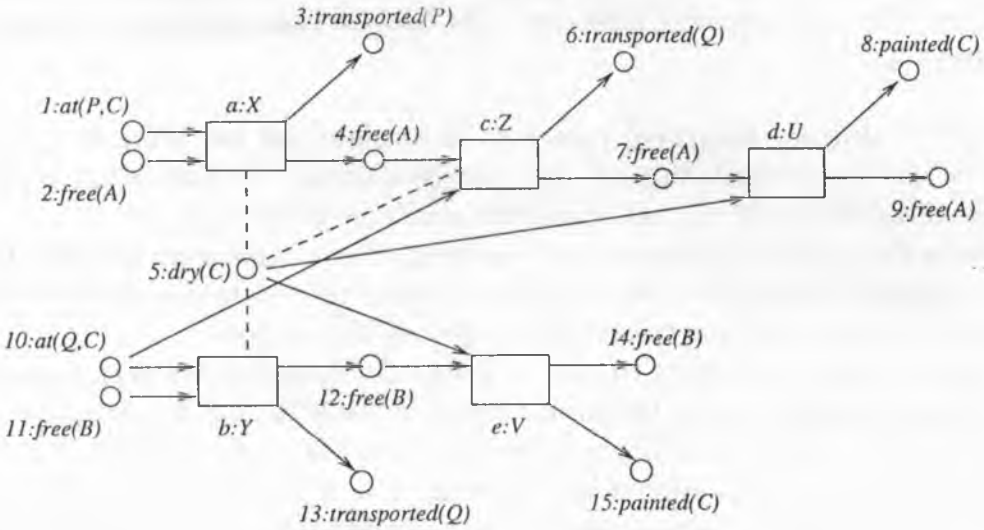

Figure 2

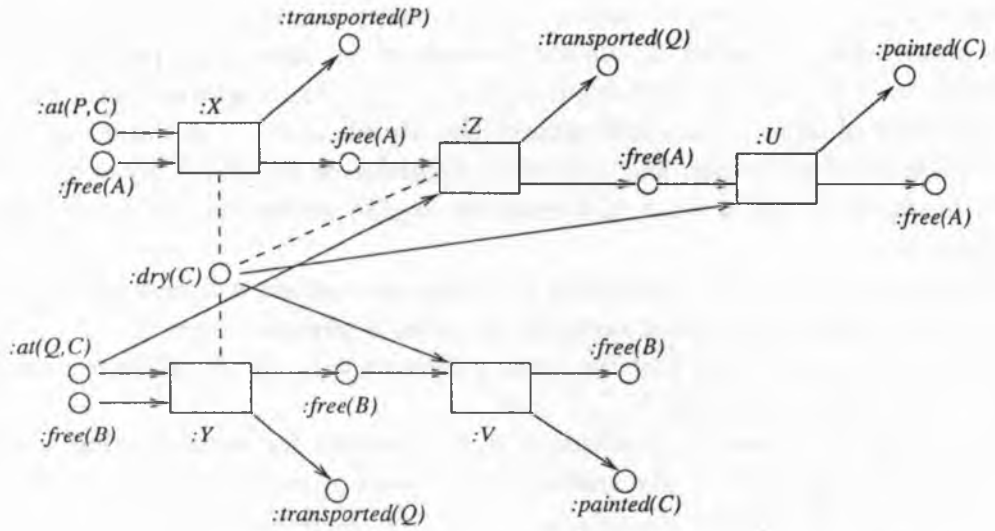

Figure 3

2 Relational rewriting systems

Systems of our concern have states represented by sets of sentences, each sentence saying that some objects are or are not in a relation, and they can change states due to actions represented by instances of rewriting rules. They are rewriting systems in the sense that each change of state is a replacement of some of the sentences that characterize this state by some other sentences, as specified by a general rewriting rule. They may also be interpreted as evolving structures in the sense of [GM 89]. According to [GM 89], an evolving structure is an abstract machine whose states are first-order structures of the same finite signature and with the same finite universes (sorts), and whose operations (transitions) are actions that change under some conditions the interpretations of some of function and relation symbols. In this paper we restrict ourselves to signatures without function symbols and represent considered systems rather as rewriting systems in the sense just described. We start with some preliminaries and next we give the needed formal definitions.

For a relation $R \subseteq X \times Y$ we often write $(x, y) \in R$ as xRy, we define the inverse as the relation $R^{-1} = \{(y, x) : xRy\}$, and for $A \subseteq X$ and $B \subseteq Y$ we define AR and RB as the sets $\{y : aRy$ for some $a \in A\}$ and $\{x : xRb$ for some $b \in B\}$, respectively, and we write $\{a\}R$ and $R\{b\}$ as aR and Rb, respectively. In particular, for a subset W of a set X with a partial order $\leq$ we have $\leq W = \{x \in X : x \leq w$ for some $w \in W\}$ and $W \leq = \{x \in X : w \leq x$ for some $w \in W\}$. For relations $R \subseteq X \times Y$ and $S \subseteq Y \times Z$ we define the composition of R and S as the relation $RS = \{(x, z) : xRy$ and ySz for some $y \in Y\}$. In particular, for mappings $f : X \to Y$ and $g : Y \to Z$ we define the composition of f and g as the mapping $fg : x \mapsto g(f(x))$. For a relation $R \subseteq X \times X$ by R^+ and R^* we denote respectively the transitive and the reflexive and transitive closure of R. For a relation $R \subseteq X \times X$ and a mapping $f : X \to Y$ we define the image of R under f as the relation $f(R) = \{(f(x), f(x')) \in Y \times Y : xRx'\}$. For a set X, by $|X|$ we denote the cardinality of X. By a *signature* we mean a triple $\Sigma = (\Omega, \Delta, arity)$, where Ω is a set of relation symbols, Δ is a set of constants, and $arity : \Omega \to \{0, 1, ...\}$ is a function that assigns to each relation symbol the number arguments of a potential relation represented by this symbol.

Let $\Sigma = (\Omega, \Delta, arity)$ be a signature and Var a set of variables.

An expression $\omega(x_1, ..., x_n)$ (resp.: $\neg\omega(x_1, ..., x_n)$), where $\omega \in \Omega$, $n = arity(\omega)$, and x_1,..., x_n are constants or variables, is called an *atomic formula*, or an *atom*, or a *positive literal* (resp.: a *negative literal*) over Σ and Var with arguments x_1,...,x_n. The negative literal $\neg\omega(x_1, ..., x_n)$ is called the *negation* of $\omega(x_1, ..., x_n)$ and each of the literals $\omega(x_1, ..., x_n)$ and $\neg\omega(x_1, ..., x_n)$ is called the *complement* of the other. For a literal L (positive or negative), by $arg(L)$ we denote the set of arguments of L. For a set S of literals by $arg(S)$ we denote the set of arguments of literals belonging to S.

A set S of literals is said to be *consistent* if it does not contain a literal and its negation.

A consistent set of literals without variables is called a *situation* over Σ.

By a *rewriting rule* over Σ and Var we mean a triple $r = (L, K, R)$ of finite consistent sets of literals over Σ and Var such that $K \subseteq L \cap R$.

Given a rewriting rule r and a substitution σ of constants for variables that is defined for all the variables occurring in r, by the *instance* of r based on σ we mean $r\sigma = (L\sigma, K\sigma, R\sigma)$, where $L\sigma$, $K\sigma$, $R\sigma$ are the results of applying σ to L, K, R, respectively.

Given a rewriting rule r, its instance $r\sigma$ based on a substitution σ, and a situation S, we say that the instance $r\sigma$ (and r itself) *applies* to S if $L\sigma \subseteq S$ and $R\sigma - L\sigma$ does not contain any literal belonging to $S - L\sigma$ or having the complement in $S - L\sigma$. In such a case $S' = (S - L\sigma) \cup R\sigma$ is a situation, we call it the result of applying $r\sigma$ (and r) to S, write $S \Rightarrow_{r\sigma} S'$, and call this triple a *direct derivation* of S' from S via $r\sigma$.

By a *derivation* we mean a finite sequence $S_1 \Rightarrow_{r_1\sigma_1} S_2 \Rightarrow_{r_2\sigma_2} S_2 ... S_n \Rightarrow_{r_n\sigma_n} S_{n+1}$ of direct derivations. If such a sequence exists for situations $S = S_1$ and $S' = S_{n+1}$, or if $S = S'$, then we say that S' is *reachable* from S.

2.1. Definition. A *relational rewriting system* (or briefly a *rewriting system*) over Σ and Var is $D = (S, \Pi, A)$, where:

(1) S is a situation over Σ (the *initial situation*),

(2) Π is a finite set of rewriting rules over Σ and Var (the set of *system rewriting rules*),

(3) A is a subset of the set of instances of the rewriting rules belonging to Π (the set of *system actions*). □

We use subscripts, S_D, Π_D, A_D when necessary.

The set S describes the initial state of the system represented by D.

Each action $a \in A$ is a triple $(pre_a, cxt_a, post_a)$ of finite consistent sets of literals without variables such that $cxt_a \subseteq pre_a \cap post_a$, called respectively the set of *preconditions*, the set of *context conditions*, and the set of *postconditions* of a. Such an action applies to each situation S in which the preconditions are satisfied and the postconditions which are not preconditions are not satisfied and have not complements which are not preconditions, and then it changes the situation such that the preconditions from $pre_a - post_a$ cease to be satisfied and the postconditions from $post_a - pre_a$ start to be satisfied. The literals belonging to cxt_a represent those preconditions which are not affected by the action and play the role of a context. As it has been noticed in [MR 95], the concept of context can be exploited to minimize the causal dependencies among applications of actions and thus to gain some concurrency. This is possible due to the fact that operations which require satisfiability of certain common literals can be applied simultaneously without disturbing each other if these common literals belong merely to their contexts.

The rewriting system D is said to be *safe* if for each situation S' that is reachable from the initial situation via actions from A, and for each action $a = (pre_a, cxt_a, post_a)$ from A, the inclusion $pre_a \subseteq S'$ implies that a applies to S'.

2.2. Example. For the system considered in the introduction we have the signature $\Sigma = (\Omega, \Delta, arity)$ with

$$\Omega = \{free, transported, dry, painted, at\},$$
$$\Delta = \{Table, A, B, C, P, Q\},$$
$$arity : free \mapsto 1, transported \mapsto 1, dry \mapsto 1, painted \mapsto 1, at \mapsto 2,$$

the set $Var = \{x, y, z\}$ of variables, and the rewriting system $D = (S, \Pi, A)$ over Σ and Var with

$$S = \{free(A), free(B), dry(C), at(P, C), at(Q.C)\},$$
$$\Pi = \{r, r'\},$$
$$r = (\{free(x), dry(z), at(y, z)\}, \{dry(z)\}, \{free(x), dry(z), transported(y)\}),$$
$$r' = (\{free(x), dry(z)\}, \emptyset, \{free(x), painted(z)\},$$

and with A being the set of instances of r and r' based on the substitutions σ such that $\sigma(x) \in \{A, B\}$, $\sigma(y) \in \{P, Q\}$, and $\sigma(z) = C$.

It is easy to check that D is safe. □

It is important to realize that safety is a property of system representation rather than of system as such. The idea of obtaining a safe representation is similar to that of transforming condition-event Petri nets into safe nets by introducing complementary conditions.

2.3. Proposition. Let $D = (S, \Pi, A)$ be any relational rewriting system over Σ and Var and let $sc(D)$ be the relational rewriting system $D' = (S', \Pi', A')$, where

(1) S' is the situation that consists of S and of the negations of those atomic formulas without variables that do not occur in S,

(2) Π' is the set of rewriting rules $r' = (L', K', R')$ that can be obtained from $r = (L, K, R) \in \Pi$ by defining K' as K, defining L' as the set of literals belonging to L or being negations of atomic formulas from $R - L$, and defining R' as the set of literals belonging to R or being negations of atomic formulas from $L - R$,

(3) A' is the set of those instances of rewriting rules from Π' which correspond to actions from A.

Then $sc(D)$ is a safe rewriting system (the *safe completion* of D). □

The proof is straightforward.

3 Contextual nets

Relational rewriting systems can be represented as contextual Petri nets in the sense of [MR 95], or as Petri nets with read arcs in the sense of [VSY 98], that is as Petri nets in which some state elements (places) play the role of a context for some transition elements (transitions). In this paper we define such nets as follows.

3.1. Definition. A *contextual net* (or briefly, a *net*) is a tuple $N = (P, T, F, C, I)$, where

(1) P is a set of *state elements* (*places*),

(2) T is a set of *transition elements* (*transitions*,) such that $P \cap T = \emptyset$,

(3) $F \subseteq P \times T \cup T \times P$ is a *flow relation* such that $Ft \neq \emptyset$ and $tF \neq \emptyset$ for all $t \in T$,

(4) $C \subseteq T \times P$ is a *context relation* such that $C \cap F = \emptyset$ and $C^{-1} \cap F = \emptyset$,

(5) $I \subseteq P$ is an *initial state* (*initial marking*). □

We denote by U the set $P \cup T$, and we use subscripts, U_N, P_N, T_N, F_N, C_N, I_N, when necessary.

Each state element $p \in P$ represents a place in which some objects, called *tokens* may appear. Each multiset s of places, that is a function $s : P \to \{0, 1, 2, ...\}$, represents a collection of tokens, $s(p)$ tokens in each place $p \in P$, called a *state* or a *marking* of N. In particular, I represents an initial collection, one token in each place $p \in I$ and no tokens in each place $p \in P - I$. Each transition element $t \in T$ represents a transition that consumes a collection of tokens, one token from each place of the set Ft, and produces a collection of tokens, one token in each place of the set tF, in the presence of a collection of tokens, one token in each place of the set Ct, the latter collection playing the role of a context that must be present when t is executed, but is not affected by the execution of t. Such a transition element is said to be *enabled* in a state s if the collections corresponding to Ft and Ct are contained in the collection represented by s, and then it transforms s to $s' = (s - Ft) + tF$.

The net N is said to be *safe* if $s(p) \leq 1$ for all places $p \in P$ and all states s that are *reachable* from the initial state I in the sense that there is a finite sequence $I = s_0, s_1, ..., s_k = s$ of states and a finite sequence $t_1, ..., t_k$ of transitions such that t_{i+1} is enabled in s_i and it transforms s_i to s_{i+1}.

For contextual nets we have the following natural notion of morphism.

3.2. Definition. A *morphism* from a net N to a net N' is a triple $m : N \to N'$, where m is a mapping from U_N to $U_{N'}$ such that

(1) $m(P_N) \subseteq P_{N'}$ and $m(T_N) \subseteq T_{N'}$,

(2) $m(F_N t) = F_{N'} m(t)$ and $m(tF_N) = m(t)F_{N'}$ and $m(C_N t) = C_{N'} m(t)$ for all $t \in T$,

(3) the restriction of m to $\{t\} \cup F_N t \cup C_N t \cup tF_N$ a bijection from $\{t\} \cup F_N t \cup C_N t \cup tF_N$ to $\{m(t)\} \cup F_{N'} m(t) \cup C_{N'} m(t) \cup m(t) F_{N'}$ for all $t \in T_N$,

(4) the restriction of m to I_N is a bijection from I_N to $I_{N'}$. □

Behaviours of contextual nets and systems represented by such nets can be defined as their branching processes in the sense of [Eng 91]. Formally, a process of a contextual net is an occurrence contextual net as defined below with a suitable morphism to the original net.

3.3. Definition. A *contextual occurrence net* (or briefly, an *occurrence net*) is a net $N = (P, T, F, C, I)$ with the following properties:

(1) for each $p \in P$ there exists at most one $t \in T$ such that tFp,

(2) $(F \cup FC)^*$, the reflexive and transitive closure of the relation $F \cup FC$, is a partial order $\leq$, called the *precedence relation*, such that each element of $U = P \cup T$ has only a finite number of predecessors with respect to $\leq$, (a *finitary* partial order),

(3) $(\leq \cup C^{-1}F)^*$, the reflexive and transitive closure of the relation $\leq \cup C^{-1}F$, is a quasi-order $\preceq$, called the *strong precedence relation*, such that, for each element of U, the restriction of $\preceq$ to the set of predecessors of this element with respect to $\leq$ is a partial order,

(4) the relation $\sharp$, where $u\sharp u'$ if $t \leq u$ and $t' \leq u'$ for $t, t' \in T$ such that $t \neq t'$ and pFt and pFt' for a some $p \in P$ (the *conflict relation*), is irreflexive, that is such that $u\sharp u'$ implies $u \neq u'$.

(5) I is min, the set of those elements of P that are minimal with respect to $\leq$. □

We use subscripts, $\leq_N$, $\prec_N$, $\sharp_N$, min_N, when necessary.

Our definition of a contextual occurrence net is similar in spirit to the definition of an occurrence net given in [VSY 98]. It differs from the latter in defining $\leq$ as the reflexive and transitive closure of the relation $F \cup FC$ rather than the reflexive and transitive closure of the relation $F \cup C$. Such a definition allows us to guarantee the important property stated in the proposition 3.9 below.

A contextual occurrence net N as defined represents a process that may branch, each branch corresponding to one of a number of possible runs of the process. Each state element $p \in P$ represents an occurrence of a fact. Each transition element $t \in T$ represents an occurrence of an action, (an *event*). For each event $t \in T$, the sets Ct, Ft, tF represent respectively the context in which t occurs and the state elements t consumes and produces. The precedence relation $\leq$ describes how state elements and events follow one another. In particular, each event that consumes a state element follows this element, each state element produced by an event follows this event, and each event that has in its context a state element produced by an event follows this event (but not the state element itself since such an element is not consumed).

Note that, according to (3) of 3.1, only state elements can be minimal w.r. to $\leq$.

The strong precedence relation $\prec$ allows to impose on each run of the represented process the requirement that there is no consumption of a state element before completing all the events having this element in the context.

3.4. Definition. A *configuration* of an occurrence net $N = (P, T, F, C, I)$ is a set $V \subseteq T$ of events of N such that:

(1) $(\leq V) \cap T \subseteq V$ (V is lower closed w.r. to the the restriction of the precedence relation $\leq$ to T),

(2) V does not contain events t and t' such that $t\sharp t'$ (V is conflict-free),

(3) the restriction of the strong precedence relation $\prec$ to the set V is a partial order (V is acyclic w.r. to the strong precedence relation $\prec$). □

Given an occurrence net N and its configuration V as defined, the set $\overline{V} = I \cup V \cup VF$ is called the *closure* of V. The restriction of N to $\overline{V}$ is an occurrence net, called the *history* corresponding to V and written as $hist(V)$. The restriction of the strong precedence relation $\prec$ to $\overline{V}$ is a partial order, and it coincides with $\prec_{hist(V)}$, the strong precedence relation of $hist(V)$. Each maximal antichain of $\overline{V}$ with respect to this partial order is called a *cut* of N. Such a cut is said to be *proper* if it does not contain events. In particular, if the set of those elements of $\overline{V}$ that are maximal with respect to $\prec_{hist(V)}$ is a maximal antichain of $\overline{V}$ with respect to $\prec_{hist(V)}$ then it is a proper cut, called the *resulting cut* of V and written as $cut(V)$.

Given a cut Z, it follows from the definitions that the set of events belonging to $\leq Z$ is a configuration. We write such a configuration as $conf(Z)$ and say that Z is *reachable* if $conf(Z)$ is finite. It is clear that $conf(cut(V)) = V$ for each configuration V that has the resulting cut, $cut(V)$.

Configurations of N are ordered by the relation: $V \sqsubseteq V'$ iff $V \subseteq V'$ and $(\preceq_{hist(V')} V) \cap T \subseteq V$, that is iff each event that is a predecessor of an event of V with respect to the strong precedence relation of $hist(V')$ belongs to V. Moreover, according to 4.1, for each event $t \in T$ there exists at least one configuration containing t, namely the configuration $(\leq t) \cap T$, and this configuration is finite and minimal in the set of configurations that contain t.

Cuts of N are ordered by the relation: $Z \Rightarrow Z'$ iff $conf(Z) \sqsubseteq conf(Z')$ and each $z \in Z$ has in Z' an upper bound z' with respect to $\preceq_{hist(conf(Z'))}$.

As the precedence relation $\leq$ is finitary, N has the least cut, namely min_N, the set of those state elements that are minimal with respect to $\leq$.

We say that N is *well bounded* if it has the greatest cut Z and coincides with $hist(conf(Z))$ for this cut. If N is well bounded then $\preceq$ is a partial order and the greatest cut is max_N, the set of maximal elements of N with respect to this order.

We say that N if *finite* if the set of state elements and events of N is finite.

The following facts allow us to consider configurations of N as partial runs of the respective process and cuts as potential stages of process development (cf. [Wink 02].

3.5. Proposition. Given a configuration V of N, and an event $v \in V$, each element of the set $Fv \cup Cv$ belongs to I or to $v'F$ with $v' \in V$ such that $v' \preceq v$ and $v' \neq v$, and it does not belong to Fv'' for any $v'' \in V$ such that $v'' \preceq v$ and $v'' \neq v$. □

3.6. Proposition. Given an event $t \in T$, and a state element $p \in Ct$, such a state element belongs to each cut of N which contains t. □

3.7. Proposition. Given a cut Z of N, the configuration $V = conf(Z)$, and an event $t \in T$, if $Ft \cup Ct \subseteq Z$ then:

(1) $V' = V \cup \{t\}$ is a configuration of N,

(2) $V \sqsubseteq V'$,

(3) $Z' = (Z - Ft) \cup tF$ is a cut of N,

(4) $conf(Z') = V'$. □

3.8. Proposition. For each reachable cut Z of N there exists a finite sequence $t_1, ..., t_n$ of events of N and a finite chain $min_N = Z_0 \Rightarrow Z_1 \Rightarrow ... \Rightarrow Z_n = Z$ of cuts such that each t_i is enabled at Z_{i-1} and it has the result Z_i in the sense that $Ft_i \cup Ct_i \subseteq Z_{i-1}$ and $conf(Z_i) = conf(Z_{i-1}) \cup \{t_i\}$. □

3.9 Proposition. Given a configuration V of N that has the resulting cut, $cut(V)$, the restriction of N to the set of those $u \in (cut(V) \leq)$ for which the context of each event $t \in (cut(V) \leq)$ such that $t \leq u$ is contained in $(cut(V) \leq)$ is an occurrence net, written as $tail_N(V)$. Moreover, for each configuration V' of N such that $V \sqsubseteq V'$, the set $V' - V$ is a configuration of $tail_N(V)$. □

The relation between an occurrence net N representing a process of a contextual net and the net itself is given by a net morphism. Consequently, a process of a contextual net N_0 is a structure that consists both of an appropriate occurrence net N and of a morphism m from this occurrence net to N_0.

3.10. Definition. Given a contextual net N_0, a *concrete branching process* (or briefly, a *concrete process*) of N_0 is a tuple $M = (P, T, F, C, I, m)$, where $onet(M) = (P, T, F, C, I)$ is an occurrence net, and $mor(M) = m$ is a net morphism such that, for all $t', t'' \in T$, the relations $Ft' = Ft''$ and $Ct' = Ct''$ and $m(t') = m(t'')$ imply $t' = t''$. □

We use subscripts, U_M, P_M, T_M, F_M, C_M, I_M, $\leq_M$, $\sharp_M$, $\preceq_M$, m_M, when necessary, and we apply to M the terminology introduced for occurrence nets.

By *configurations* (resp.: *histories*, *cuts*) of M we mean configurations (resp.: histories, cuts) of $onet(M)$. By min_M (resp.: max_M) we denote $min_{onet(M)}$ (resp.: $max_{onet(M)}$). For each cut Z of M by $tail_M(Z)$ we mean $tail_{onet(M)}(Z)$. We say that M is *well bounded* (resp.: *finite*) if such is $onet(M)$.

The condition imposed on the morphism m is exactly as in [Eng 91]. Note that each contextual net has a concrete process, namely the process that consists of the restriction of this net to its initial state and of the embedding of this restriction into the net.

Let N_0 be a contextual net.

3.11. Definition. A *process morphism* (or briefly, a *morphism*) from a concrete process M of N_0 to a concrete process M' of N_0 is a net morphism $h : onet(M) \to onet(M')$ such that $hm_{M'} = m_M$. □

3.12. Definition. An *abstract branching process* (or briefly, an *abstract process*, or a *process*) of N_0 is an isomorphism class of concrete processes of N_0. □

3.13. Definition. Given two abstract processes μ and μ' of N_0, the process μ is said to *approximate* the process μ', written as $\mu \trianglelefteq_{N_0} \mu'$, if there exists an injective morphism from a concrete process $M \in \mu$ to a concrete process $M' \in \mu'$. □

3.14. Theorem. The relation $\trianglelefteq_{N_0}$ is a partial order. □

3.15. Theorem. The set of abstract processes of N_0 with the partial order $\trianglelefteq_{N_0}$ is a complete lattice. □

By $Processes(N_0)$ we denote the lattice of abstract processes of N_0 with the partial order $\trianglelefteq_{N_0}$.

3.16. Definition. The *unfolding* (or the *behaviour*) of N_0 is the greatest element of the lattice $Processes(N_0)$, written as $unf(N_0)$. Each concrete process in the isomorphism class $unf(N_0)$ is called a *concrete unfolding* of N_0. □

3.17. Proposition. A contextual net N_0 is *safe* iff each concrete process $M = (P, T, F, C, I, m)$ of N_0 enjoys the following property: for all $p', p'' \in P$ such that $p' \neq p''$, the equality $m(p') = m(p'')$ implies that there is no cut of M containing both p' and p''. □

In the rest of this section we restrict ourselves to safe contextual nets and we recall the respective concepts and facts presented in [Wink 02].

Let N_0 be a finite safe contextual net.

3.18. Definition. Given a cut Z of the underlying occurrence net of a concrete process M of

a safe contextual net N_0, the *state* or the *marking* of M at Z, $state_M(Z)$, is defined as follows:

$$state_M(Z) = \{mor_M(z) : z \in Z\}. \square$$

Due to 3.5, 3.8, and 3.1, this definition is consistent with the standard definition of reachable markings.

3.19. Proposition. A set $s \subseteq P_{N_0}$ is a reachable state of N_0 iff there exists a reachable cut Z of a concrete process M of N_0 such that $s = state_M(Z)$. $\square$

The unfolding of a safe contextual net contains information about all reachable states of this net. In this section we show that if a net is also finite then this information is contained in a finite prefix of the unfolding, and we adapt the known algorithm of McMillan of constructing such a prefix (cf. [McM 93]). Due to the specific definitions of the relations of precedence and strong precedence in contextual occurrence nets, these results do not need any particular restrictions of the class of nets as in [VSY 98].

3.20. Definition. Given a concrete unfolding M of N_0, a configuration V of M is said to be *prime* if it has a unique event t that is maximal w.r. to the restriction to V of the strong precedence relation of $\preceq_M$. The set of prime configurations with the unique maximal event t is denoted by $[t]$. $\square$

3.21. Proposition. Each prime configuration of the underlying occurrence net of a concrete unfolding of N_0 is finite. $\square$

3.22. Definition. A prime configuration of a concrete unfolding of N_0 is said to be a *cut-off configuration* of M if it contains two different prime subconfigurations V' and V'' such that $V' \sqsubseteq V''$ and $state_M(cut(V')) = state_M(cut(V''))$. $\square$

3.23. Definition. An event t of a concrete unfolding of N_0 is said to be a *cut-off event* (resp.: an *informative event*) if each prime configuration $V \in [t]$ is a cut-off configuration (resp.: there exists a prime $V \in [t]$ that is not a cut-off configuration). $\square$

3.24. Theorem. Given a concrete unfolding M of N_0, the set of informative events of M, written as $inf(M)$, is finite. Moreover, the restriction of M to the set $I \cup inf(M) \cup inf(M)F_M$, written as $B(M)$, is a *prefix* of M in the sense that it approximates M. $\square$

3.25. Theorem. Given a concrete unfolding M of N_0, the prefix $B(M)$ of M is *complete* in the sense that for each reachable state s of N_0 there exists a configuration V of $B(M)$ such that $state_{B(M)}(cut(V)) = s$. $\square$

The construction of a finite complete prefix of the unfolding of a finite safe contextual net results in a concrete process of the net with a finite set of maximal configurations. Consequently, in order to find out how to reach a state with given contents of certain places it suffices to investigate the subconfigurations of the maximal configurations of the finite complete prefix. In this section we show that it is possible with a modified version of the method of Esparza of representing the problem as a problem of linear programming (cf. [Espa 93]).

Let N_0 be a finite safe contextual net. By modifying the ideas of [Espa 93] one can obtain the following results (cf. [Wink 02]).

3.26. Proposition. Given a concrete unfolding M of N_0, its configuration V, and a nonempty family $(V_k : k \in K)$ of subconfigurations of V, the set $\bigcup(V_k : k \in K)$ is a configuration of M and it is the least upper bound of the family $(V_k : k \in K)$ with respect to the partial order $\sqsubseteq$. □

3.27. Proposition. Given a concrete unfolding M of N_0, its configuration V, and two disjoint sets S^+ and S^- of state elements of N_0, if the set of subconfigurations V' of V such that $S^+ \subseteq state_M(cut(V'))$ and $S^- \cap state_M(cut(V')) = \emptyset$, written as $Last(S^+, S^-, V)$, is nonempty, then it contains the greatest member, that is the greatest subconfiguration V' of V such that $S^+ \subseteq state_M(cut(V'))$ and $S^- \cap state_M(cut(V')) = \emptyset$. □

3.28. Theorem. Given a concrete unfolding M of N_0, its configuration V, and two disjoint sets S^+ and S^- of state elements of N_0, if the set $Last(S^+, S^-, V)$ is nonempty, W is the greatest subconfiguration of V that belongs to this set, and $X : V \to \{0, 1\}$ is the characteristic function of W as a subset of V, that is $X(v) = 1$ for $v \in W$ and $X(v) = 0$ for $v \in V - W$, then the values of this function constitute a solution of the linear programming problem of finding maximum of $\Sigma(X(v) : v \in V)$ such that the following conditions are satisfied:

(1) for every $v \in V$: $0 \leq X(v) \leq 1$,

(2) for every $p \in FV \cap VF$: $X(pF) \leq X(Fp)$,

where pF denotes the unique $u \in V$ such that pFu, and Fp denotes the unique $v \in V$ such that vFp,

(3) for every $s \in S^+$: $\Sigma(Y(p) : p \in H(s)) = 1$, where

$$Y(p) = \begin{cases} 1 & \text{if } p \in min_M - FV \\ 1 - X(pF) & \text{if } p \in min_M \cap FV \\ X(Fp) & \text{if } p \in VF - FV \\ X(Fp) - X(pF) & \text{if } p \in FV \cap VF \end{cases}$$

and where $H(s)$ is the set of state elements p of $min_M \cup VF$ with $(mor(M))(p) = s$,

(4) for every $s \in S^-$: $\Sigma(Y(p) : p \in H(s)) = 0$,

(5) for every $u, v \in V$ such that uFp and pCv for some p: $X(v) \leq X(u)$,

(6) for every $u, v \in V$ such that pCu and pFv for some p: $X(v) \leq X(u)$. □

3.29. Theorem. Given a concrete unfolding M of N_0, its configuration V, and two disjoint sets S^+ and S^- of state elements of N_0, if the linear programming problem as in 3.28 has no solution then the set $Last(S^+, S^-, V)$ is empty. Otherwise this problem has a unique solution and this solution is the set of values of the characteristic function of the greatest configuration of $Last(S^+, S^-, V)$. □

4 Safe relational rewriting systems as contextual nets

There is a natural correspondence between safe rewriting systems and safe contextual nets. This allows to represent the behaviours of rewriting systems by the behaviours of the corresponding nets.

Let $\Sigma = (\Omega, \Delta, arity)$ be a signature and Var a set of variables.

4.1 Definition. Given a safe relational rewriting system $D = (S, \Pi, A)$ over Σ and Var, the *net corresponding to* D (or briefly, the *net of* D) is the net $net(D) = (P, T, F, C, I)$, where:

(1) P is a set of literals without variables over Σ (a set of *facts*),

(2) T is the set of actions of D, that is $T = A$,

(3) $F = \{(p,t) \in P \times T : p \in pre_t - cxt_t\} \cup \{(t,p) \in T \times P : p \in post_t - cxt_t\}$,

(4) $C = \{(p,t) \in P \times T : p \in cxt_t\}$,

(5) $I = S$. □

We use subscripts, P_D, T_D, F_D, C_D, I_D, when necessary. Thus $P_D = P_{net(D)}$, $T_D = T_{net(D)}$, $F_D = F_{net(D)}$, $C_D = C_{net(D)}$, $I_D = I_{net(D)}$.

4.2. Proposition. All reachable states of contextual nets corresponding to safe rewriting systems are situations. Moreover, contextual nets corresponding to safe rewriting systems are safe. □

Proof outline: Each transition t of the net corresponding to a rewriting system $D = (S, \Pi, A)$ can be enabled in a state s that is a situation only if $pre_t \subseteq s$. As D is safe, the inclusion $pre_t \subseteq s$ implies that t is enabled as an action of D and transforms s into a situation s'. On the other hand, I is a situation. Consequently, each reachable state of $net(D)$ is a situation. In particular, each reachable state of $net(D)$ does not contain more than one copy of a literal. Thus $net(D)$ is a safe contextual net. □

Processes of safe rewriting systems can be defined as processes of the corresponding contextual nets.

4.3. Definition. Given a safe rewriting system D, a *concrete branching process* (or briefly, a *concrete process*) of D is a concrete process of the corresponding contextual net $net(D)$. □

From 4.2 and 3.8 we obtain the following proposition.

4.4. Proposition. Given a concrete process $M = (P, T, F, C, I, m)$ of a safe rewriting system D, for all $p', p'' \in P$ such that $p' \neq p''$, each of the equalities $m(p') = m(p'')$, $m(p') = \neg m(p'')$, $\neg m(p') = m(p'')$ implies that there is no cut of $net(M)$ containing both p' and p''. □

4.5. Conclusion. Given a concrete unfolding M of a safe rewriting system D and a reachable cut Z of M, the state of M at Z, $state_M(Z)$, is a consistent set of literals. □

4.6. Definition. An *abstract branching process* (or briefly, an *abstract process*, or a *process*) of a rewriting system D is an isomorphism class of concrete processes of D. □

4.7. Definition. The *behaviour* or the *unfolding* of a safe rewriting system D is the greatest element of the lattice $Processes(net(D))$, that is $unf(net(D))$. Each concrete process in the isomorphism class $unf(net(D))$ is called a *concrete behaviour* or a *concrete unfolding* of D. □

The results about safe contextual nets apply to safe rewriting systems. From 3.24, 3.28 and 3.29 we obtain results which allow to solve the problem of reachability for safe rewriting systems.

Let D be a safe rewriting system.

4.8. Proposition. Given a concrete unfolding M of D, its configuration V, and two disjoint sets S^+ and S^- of literals without variables, if the set of subconfigurations V' of V such that $S^+ \subseteq state_M(cut(V'))$ and $S^- \cap state_M(cut(V')) = \emptyset$, written as $Last(S^+, S^-, V)$, is nonempty, then it contains the greatest member, that is the greatest subconfiguration V' of V such that $S^+ \subseteq state_M(cut(V'))$ and $S^- \cap state_M(cut(V')) = \emptyset$. □

4.9. Theorem. Given a concrete unfolding M of D, its configuration V, and two disjoint sets S^+ and S^- of literals without variables, if the set $Last(S^+, S^-, V)$ is nonempty, W is the greatest subconfiguration of V that belongs to this set, and $X : V \to \{0, 1\}$ is the characteristic function of W as a subset of V, that is $X(v) = 1$ for $v \in W$ and $X(v) = 0$ for $v \in V - W$, then the values of this function constitute a solution of the linear programming problem of finding maximum of $\Sigma(X(v) : v \in V)$ such that the following conditions are satisfied:

(1) for every $v \in V$: $0 \leq X(v) \leq 1$,

(2) for every $p \in FV \cap VF$: $X(pF) \leq X(Fp)$,

where pF denotes the unique $u \in V$ such that pFu, and Fp denotes the unique $v \in V$ such that vFp,

(3) for every $s \in S^+$: $\Sigma(Y(p) : p \in H(s)) = 1$, where

$$Y(p) = \begin{cases} 1 & \text{if } p \in min_M - FV \\ 1 - X(pF) & \text{if } p \in min_M \cap FV \\ X(Fp) & \text{if } p \in VF - FV \\ X(Fp) - X(pF) & \text{if } p \in FV \cap VF \end{cases}$$

and where $H(s)$ is the set of state elements p of $min_M \cup VF$ with $(mor(M))(p) = s$,

(4) for every $s \in S^-$: $\Sigma(Y(p) : p \in H(s)) = 0$,

(5) for every $u, v \in V$ such that uFp and pCv for some p: $X(v) \leq X(u)$,

(6) for every $u, v \in V$ such that pCu and pFv for some p: $X(v) \leq X(u)$. □

4.10. Theorem. Given a concrete unfolding M of D, its configuration V, and two disjoint sets S^+ and S^- of literals without variables, if the linear programming problem as in 4.9 has no solution then the set $Last(S^+, S^-, V)$ is empty. Otherwise this problem has a unique solution and this solution is the set of values of the characteristic function of the greatest configuration of $Last(S^+, S^-, V)$. □

The following example illustrates an application of these results.

4.11. Example. The safe rewriting system in 2.2 has the net shown in figure 1. The finite complete prefix of the unfolding of this net is just the entire unfolding. According to the results

just stated the characteristic function of a configuration resulting in a marking corresponding to the state

$$S' = \{transported(P), transported(Q), painted(C), free(A), free(B)\}$$

in the upper branch of the net in figure 2 should satisfy the following conditions:

$$0 \leq X(a), X(c), X(d) \leq 1$$

$$X(d) \leq X(c) \leq X(a)$$

$$X(a) = X(c) = X(d) = 1$$

$$(1 - X(a)) + (X(a) - X(c)) + (X(c) - X(d)) + X(d) = 1$$

Consequently, it must be the configuration $\{a, c, d\}$.
Similarly, for the other branch we obtain the following conditions:

$$0 \leq X(b), X(e) \leq 1$$

$$X(e) \leq X(b)$$

$$X(b) = X(e) = 1$$

$$\Sigma(Y(p) : label(p) = transported(P)) = 1$$

with $\Sigma(Y(p) : label(p) = transported(P)) = 0$ (due to the lack of p such that $label(p) = transported(P)$). Consequently, the branch does not contain a marking corresponding to S'. □

5 Recapitulation

In problems of planning one has to do with the problem of finding a way of reaching in a system a state or a class of states. In order to solve such a problem one has to describe the behaviour of the considered system.

In the case of systems with considerable independence of actions the description of behaviour can essentially be reduced by representing systems as safe Petri nets and by representing system behaviours by fragments of unfoldings of such nets (cf. [McM 93]).

Taking into account the fact that some conditions of executing actions play only the role of a context, one can further reduce the descriptions of systems and their behaviours using safe contextual nets (cf. [MR 95], [VSY 98], etc.).

However, the gain in the description of system behaviour by standard or contextual safe Petri nets has a price since the global system states are represented in such a description only implicitly - as cuts of the respective occurrence nets. So, an effective method of searching for states with suitable properties is needed. For standard Petri nets such a method has been proposed in [Espa 93]. For safe contextual nets one can use the modified version of this method described in [Wink 02].

All the above elements of problem solution have been described in [Wink 01] and [Wink 02] and summarized in the present paper. When developed and endowed with an obvious control mechanism, they can easily be combined into an algorithm for planning.

References

[Drum 89] Drummond, M., *Situated Control Rules*, in Proc. of the First Int. Conf. on Principles of Knowledge Representation and Reasonning, Toronto, May 1989, 103-113

[Eng 91] Engelfriet, J., *Branching Processes of Petri Nets*, Acta Informatica 28 (1991) 575-591

[EPS 73] Ehrig, H., Pfender, H., Schneider, H. J., *Graph Grammars: An Algebraic Approach*, Proc. of the IEEE Conf. on Automata and Switching Theory, Iowa City (1973) 167-180

[Espa 93] Esparza, J., *Model Checking Using Net Unfoldings*, in Proc. of TAPSOFT'93, Springer LNCS 668 (1993) 613-628

[GM 89] Gurevich, Y., Moss, L. S., *Algebraic Operational Semantics and Occam*, in the Proceedings of CSL'89, Kaiserslautern, October 1989, Springer LNCS 440 (1989) 176-192

[GoKa 91] Godefroid, P., Kabanza, F., *An Efficient Reactive Planner for Synthesizing Reactive Plans*, in Proc. of AAAI'91 (1991) 640-645

[Kowa 76] Kowalski, R., *Logic for Problem Solving*, North Holland, New York (1979)

[MaWi 82] Maggiolo-Schettini, A., Winkowski, J., *Processes of Transforming Structures*, J. Comput. System Sci., Vol.24, No.3 (1982) 245-282

[McM 93] McMillan, K., L., *Using unfoldings to avoid the state explosion problem in the verification of asynchronous circuits*, in Proc. of the 4th Workshop on Computer Aided Verification, Montreal 1992, G. v. Bochmann and D. K. Probst (Eds.), Springer LNCS 663 (1993) 164-174

[MR 95] Montanari, U., Rossi, F., *Contextual Nets*, Acta Informatica 32 (1995) 545-596

[VSY 98] Vogler, W., Semenov, A., Yakovlev, A., *Unfolding and Finite Prefix for Nets with Read Arcs*, Proc. of CONCUR'98, Nice, September 1998, Davide Sangiorgi and Robert de Simone (Eds.), Springer LNCS 1466 (1998) 501-516

[Wink 01] Winkowski, J., *A model of evolving relational structures and its application to plan-formation*, ICS PAS Report 929 (2001)

[Wink 02] Winkowski, J., *Reachability in contextual nets*, Fundamenta Informaticae 51 (2002) 235-250

Pracę zgłosił: Antoni Mazurkiewicz

Adres autora Jozef Winkowski
Instytut Podstaw Informatyki PAN
Ordona 21
01-237 Warszawa
e-mail: wink@ipipan.waw.pl

Symbole klasyfikacji rzeczowej F.1.1, F.1.2, I.2.8
Na prawach rękopisu
Printed as a manuscript

Nakład 140 egzemplarzy. Papier kserograficzny klasy III. Oddano do druku we wrześniu 2002. Wydawnictwo IPI PAN. ISSN 0138-06-48.

ISSN 0138-0648