# Process Ordering in a Process Calculus for Spatially-Explicit Ecological Models

Anna Philippou and Mauricio Toro(✉)

Department of Computer Science, University of Cyprus, Nicosia, Cyprus
{annap,mtoro}@cs.ucy.ac.cy

**Abstract.** In this paper we extend PALPS, a process calculus proposed for the spatially-explicit individual-based modeling of ecological systems, with the notion of a *policy*. A policy is an entity for specifying orderings between the different activities within a system. It is defined externally to a PALPS model as a partial order which prescribes the precedence order between the activities of the individuals of which the model is comprised. The motivation for introducing policies is twofold: one the one hand, policies can help to reduce the state-space of a model; on the other hand, they are useful for exploring the behavior of an ecosystem under different assumptions on the ordering of events within the system. To take account of policies, we refine the semantics of PALPS via a transition relation which prunes away executions that do not respect the defined policy. Furthermore, we propose a translation of PALPS into the probabilistic model checker PRISM. We illustrate our framework by applying PRISM on PALPS models with policies for conducting simulation and reachability analysis.

## 1 Introduction

Population ecology is a sub-field of ecology that deals with the dynamics of species populations and their interactions with the environment. Its main aim is to understand how the population sizes of species change over time and space. It has been of special interest to conservation scientists and practitioners who are interested in predicting how species will respond to specific management schemes and in guiding the selection of reservation sites and reintroduction efforts, e.g. [12,21].

One of the main streams of today's theoretical ecology is the *individual-based* approach to modeling population dynamics. In this approach, the modeling unit is that of a *discrete individual* and a system is considered as the composition of individuals and their environment. Since individuals usually move from one location to another, it is common in individual-based modeling to represent

This work was carried out during the tenure by the second author of an ERCIM "Alain Bensoussan" Fellowship Programme. The research leading to these results has received funding from the European Union Seventh Framework Programme (FP7/2007-2013) under grant agreement no 246016.

S. Counsell and M. Núñez (Eds.): SEFM 2013 Collocated Workshops, LNCS 8368, pp. 345–361, 2014.
DOI: 10.1007/978-3-319-05032-4_25, 

space explicitly. There are four distinct frameworks in which *spatially-explicit individual-based models* can be defined [4] which differ on whether space and time are treated discretely or continuously. The four resulting frameworks have been widely studied in Population ecology and they are considered to complement as opposed to compete with each other.

In this paper, we extend our previous work on a process-calculus framework for the spatially-explicit modeling of ecological systems. Our process calculus, PALPS, follows the individual-based modeling and, in particular, it falls in the discrete-time, discrete-space class of Berec's taxonomy [4]. The extension presented in this paper is related to the issue of *process ordering*. In particular, simulations carried out by ecologists often impose an order on the events that may take place within a model. For instance, if we consider mortality and reproduction within a single-species model, three cases exist: concurrent ordering, reproduction preceding mortality and reproduction following mortality. In *concurrent ordering*, different individuals may reproduce and die simultaneously. For *reproduction preceding mortality*, the population first reproduces, then all individuals, including new offspring, are exposed to death. For *reproduction following mortality*, individuals are first exposed to death and, subsequently, surviving individuals are able to reproduce. Ordering can have significant implications on the simulation. Thus, alternatives must be carefully studied so that the ordering that best matches the observed behavior of the actual ecosystem can be concluded (see e.g. [27]).

In order to capture process ordering in PALPS, we define the notion of a *policy*, an entity that imposes an order on the various events that may take place within a system. Formally, a policy $\sigma$ is defined as a partial order on the set of events in the system where, by writing $(\alpha, \beta) \in \sigma$, we specify that whenever there is a choice between executing the activities $\alpha$ and $\beta$, $\beta$ is chosen. As a result, a policy is defined externally to a process description. This implies that one may investigate the behavior of a system under different event orderings simply by redefining the desired policy without redeveloping the system's description. To capture policies in the semantics of PALPS we extend its transition relation into a *prioritized* transition relation which prunes away all transitions that do not respect the defined policy.

Furthermore, we present a methodology for analyzing models of PALPS *with policies* via the probabilistic model checker PRISM [1]. To achieve this, we define a translation of PALPS *with policies* into the PRISM language and we prove its correctness. We then apply our methodology on simple examples and we demonstrate the types of analysis that can be performed on PALPS models via the PRISM tool. By contrasting our results with our previous work of [24], we observe that policies may achieve a significant reduction in the size of models and may thus enable the analysis of larger systems.

Various formal frameworks have been proposed in the literature for modeling biological and ecological systems. One strand is based, like PALPS, on process calculi, and constitute extensions of calculi such as CCS [17], the $\pi$-calculus [18] and CSP [13]. Examples include [6,9,15,26,29]. A different approach is that of

*P systems* [25], conceived as a class of distributed and parallel computing inspired by the compartmental structure and the functioning of living cells. P-systems have been extended in various directions and they have been applied to a wide range of applications including the field of ecology [5,7,20,22]. Finally, we mention the calculus of looping sequences [3], and its spatial extension [2], *synchronous automata* [11] and *cellular automata* [8]. Similarly to ecosystem modeling carried out by Ecologists, these approaches differ in their treatment of time and space and can be considered as supplements as opposed to rivals as each offers a distinct view and different techniques for analyzing systems. In particular, the discrete-time approach, also adopted in PALPS, has proved useful and appropriate for studying predator-prey systems, epidemiology systems, and for studying population dynamics in other types of ecological systems in frameworks such as WSCCS [14] and P-Systems [5,7].

Regarding the notion of policies employed in PALPS *with policies*, we point out that they are essentially a type of priorities usually referred to in the process-algebra literature as *static priority relations* (see e.g. [10]) and are similar to the priorities defined for P-Systems. In comparison to related works, as far as we know, PALPS *with policies* is the first spatially-explicit formalism for ecological systems that includes the notion of priority and employs this notion to experiment with different process orderings within process descriptions. Furthermore, via the translation to the PRISM language our framework enables to carry out more advanced analysis of ecological models than just simulation, which is the main approach adopted in the related literature. Possible analysis techniques are those supported by the PRISM tool and include model-checking, reachability analysis as well as computing expected behavior.

*Structure of the paper.* The structure of the remainder of the paper is as follows. In Sect. 2 we present the syntax and the semantics of PALPS *with policies* and we illustrate the expressiveness of the calculus via a number of examples. In Sect. 3 we sketch a translation of PALPS into the Markov-decision-process component of the PRISM language. We then apply our methodology on simple examples and we explore the potential of the approach in Sect. 4. Section 5 concludes the paper. For the full exposition of the PRISM translation and its correctness proof the reader is referred to [23].

## 2 The Process Calculus

In this section, we extend our previous work on the process calculus PALPS with the notion of a *policy*. A policy is an entity that accompanies the process description of a system and specifies precedence orders between the various activities that may take place within the system. In this section we review the syntax and semantics of PALPS, and we describe how policies are defined and how they affect the semantics of the framework. We illustrate the calculus via simple examples. For a more thorough presentation of PALPS and further examples the reader is referred to [23,24].

### 2.1 The Syntax

In PALPS, we consider a system as a set of individuals operating in space, each belonging to a certain species and inhabiting a location. This location may be associated with attributes which describe characteristics of the location and can be used to define location-dependent behavior of individuals. Furthermore, individuals who reside at the same location may communicate with each other upon channels, e.g. for preying, or they may migrate to a new location. PALPS models probabilistic events with the aid of a probabilistic operator and uses a discrete treatment of time.

The syntax of PALPS is based on the following basic entities: (1) $\mathbf{S}$ is a set of species ranged over by $\mathbf{s}$, $\mathbf{s}'$. (2) $\mathbf{Loc}$ is a set of locations ranged over by $\ell$, $\ell'$. The habitat of a system is then implemented via a relation $\mathbf{Nb}$, where $(\ell, \ell') \in \mathbf{Nb}$ exactly when locations $\ell$ and $\ell'$ are neighbors. For convenience, we use $\mathbf{Nb}$ as a function and write $\mathbf{Nb}(\ell)$ for the set of all neighbors of $\ell$. (3) $\mathbf{Ch}$ is a set of channels ranged over by lower-case strings. (4) $\Psi$ is a set of attributes, ranged over by $\psi$, $\psi'$. We write $\psi_\ell$ for the value of attribute $\psi$ at location $\ell$. Attributes may capture characteristics of a location e.g. its capacity or its temperature.

PALPS employs two sets of expressions: *logical expressions*, ranged over by $e$, and *arithmetic expressions*, ranged over by $w$. They are constructed as follows

$$\begin{aligned} e &::= true \mid \neg e \mid e_1 \wedge e_2 \mid w \bowtie c \\ w &::= c \mid \psi@\ell^\star \mid \mathbf{s}@\ell^\star \mid \mathbf{op}_1(w) \mid \mathbf{op}_2(w_1, w_2) \end{aligned}$$

where $c$ is a real number, $\bowtie \in \{=, \leq, \geq\}$, $\ell^\star \in \mathbf{Loc} \cup \{\mathsf{myloc}\}$ and $\mathbf{op}_1$ and $\mathbf{op}_2$ are the usual unary and binary arithmetic operations on real numbers. Expression $\psi@\ell^\star$ denotes the value of attribute $\psi$ at location $\ell^\star$ and expression $\mathbf{s}@\ell^\star$ denotes the number of individuals of species $\mathbf{s}$ at location $\ell^\star$. In the case that $\ell^\star = \mathsf{myloc}$, then the expression refers to the value of $\ell$ at the actual location of the individual in which the expression appears and it is instantiated to this location when the condition needs to be evaluated.

The syntax of PALPS is given at three levels: (1) the individual level ranged over by $P$, (2) the species level ranged over by $R$, and (3) the system level ranged over by $S$. Their syntax is defined via the following BNFs:

$$\begin{aligned} P &::= \mathbf{0} \mid \sum_{i \in I} \eta_i.P_i \mid \sum_{i \in I} p_i{:}P_i \mid \mathsf{cond}\,(e_1 \rhd P_1, \ldots, e_n \rhd P_n) \mid C \\ R &::= !rep.P \\ S &::= \mathbf{0} \mid P{:}\langle \mathbf{s}, \ell \rangle \mid R{:}\langle \mathbf{s} \rangle \mid S_1 \,|\, S_2 \mid S \backslash L \end{aligned}$$

where $L \subseteq \mathbf{Ch}$, $I$ is an index set, $p_i \in (0, 1]$ with $\sum_{i \in I} p_i = 1$, $C$ ranges over a set of process constants $\mathcal{C}$, each with an associated definition of the form $C \stackrel{\text{def}}{=} P$, and

$$\eta ::= a \mid \overline{a} \mid go\ \ell \mid \surd .$$

Beginning with the individual level, $P$ can be one of the following: Process $\mathbf{0}$ represents the inactive individual, that is, an individual who has ceased to

exist. Process $\sum_{i\in I} \eta_i.P_i$ describes the non-deterministic choice between a set of action-prefixed processes. We write $\eta_1.P_1+\eta_2.P_2$ to denote the binary form of this operator. In turn, an activity $\eta$ can be an input action on a channel $a$, written simply as $a$, a complementary output action on a channel $a$, written as $\overline{a}$, a movement action with destination $\ell$, $go\,\ell$, or the time-passing action, $\surd$. Actions of the form $a$, and $\overline{a}$, $a \in \mathbf{Ch}$, are used to model arbitrary activities performed by an individual; for instance, eating, preying and reproduction. The tick action $\surd$ measures a tick on a global clock. These time steps are abstract in the sense that they have no defined length and, in practice, $\surd$ is used to separate the rounds of an individual's behavior. Process $\sum_{i\in I} p_i{:}P_i$ represents the probabilistic choice between processes $P_i$, $i \in I$. The process randomly selects an index $i \in I$ with probability $p_i$, and then evolves to $P_i$. We write $p_i{:}P_i \oplus p_2{:}P_2$ for the binary form of this operator. The conditional process $\mathsf{cond}\ (e_1 \rhd P_1, \ldots, e_n \rhd P_n)$ presents the conditional choice between a set of processes: it behaves as $P_i$, where $i$ is the smallest integer for which $e_i$ evaluates to $\mathsf{true}$. Note that this choice is deterministic. Finally, process constants provide a mechanism for including recursion in the calculus.

Moving on to the species level, we employ the special *species process* $R$ defined as $!rep.P$. This is a replicated process which may always receive input through channel $rep$ and create new instances of process $P$, where $P$ is a new individual of species $R$.

Finally, population systems are built by composing in parallel located individuals and species. An individual is defined as $P{:}\langle \mathbf{s}, \ell\rangle$, where $\mathbf{s}$ and $\ell$ are the species and the location of the individual, respectively. A species is given by $R{:}\langle \mathbf{s}\rangle$, where $\mathbf{s}$ is the name of the species. In a composition $S_1 \mid S_2$ the components may proceed independently on their channel-based actions or synchronize with one another while executing complementary actions, in the CCS style, and they must synchronize on their $\surd$ actions. Essentially, the intention is that, in any given round of the lifetime of the individuals, all individuals perform their available actions possibly synchronizing as necessary until they synchronize on their next $\surd$ action and proceed to their next round. Finally, $S\backslash L$ models the restriction of the channels in set $L$ within $S$. As a syntactic shortcut, we will write $P{:}\langle \mathbf{s}, \ell, n\rangle$ for the parallel composition of $n$ copies of process $P{:}\langle \mathbf{s}, \ell\rangle$.

### 2.2 The Unprioritized Semantics

The semantics of PALPS is defined in terms of a *structural operational semantics* given at the level of configurations of the form $(E, S)$, where $E$ is an *environment* and $S$ is a population system. The environment $E$ is an entity of the form $E \subset \mathbf{Loc} \times \mathbf{S} \times \mathbb{N}$, where each pair $\ell$ and $\mathbf{s}$ is represented in $E$ at most once and where $(\ell, s, m) \in E$ denotes the existence of $m$ individuals of species $s$ at location $\ell$. The environment $E$ plays a central role in evaluating expressions.

Initially, we define the *unprioritized* semantics of PALPS. This semantics is then refined into the *prioritized semantics* which takes into account the notion of policies in Sect. 2.3. The unprioritized semantics is given in terms of two transition relations: the non-deterministic relation $\longrightarrow_n$ and the probabilistic relation

$\longrightarrow_p$. A transition of the form $(E, S) \xrightarrow{\alpha}_n (E', S')$ means that a configuration $(E, S)$ may execute action $\alpha$ and become $(E', S')$. A transition of the form $(E, S) \xrightarrow{w}_p (E', S')$ means that a configuration $(E, S)$ may evolve into configuration $(E', S')$ with probability $w$. Action $\alpha$ appearing in the non-deterministic relation may have one of the following forms:

- $a_{\ell,\mathbf{s}}$ and $\overline{a}_{\ell,\mathbf{s}}$ denote the execution of actions $a$ and $\overline{a}$ respectively at location $\ell$ by an individual of species $\mathbf{s}$.
- $\tau_{a,\ell,\mathbf{s}}$ denotes an internal action that has taken place on channel $a$, at location $\ell$, and where the output on $a$ was carried out by an individual of species $\mathbf{s}$. This action may arise when two complementary actions take place at the same location $\ell$ or when a move action take place from location $\ell$.

Due to the space limitations, the rules of the PALPS semantics are omitted. The interested reader may refer to [23] for the details.

### 2.3 Policies and Prioritized Semantics

We are now ready to define the notion of a policy and refine the semantics of PALPS accordingly. A policy $\sigma$ is a partial order on the set of PALPS non-probabilistic actions. By writing $(\alpha, \beta) \in \sigma$ we imply that action $\beta$ has higher priority than $\alpha$ and whenever there is a choice between $\alpha$ and $\beta$, $\beta$ should always be selected. For example, the policy $\sigma = \{(\text{reproduce}_{\ell,\mathbf{s}}, \text{disperse}_{\ell,\mathbf{s}}) | \ell \in \mathbf{Loc}\}$ specifies that, at each location, dispersal actions of species $\mathbf{s}$ should take place before reproduction actions. On the other hand $\sigma = \{(\text{reproduce}_{\ell_1,\mathbf{s}}, \text{disperse}_{\ell_1,\mathbf{s}}), (\text{disperse}_{\ell_2,\mathbf{s}}, \text{reproduce}_{\ell_2,\mathbf{s}})\}$ specifies that, while dispersal should proceed reproduction at location $\ell_1$, the opposite should hold at location $\ell_2$.

To achieve this effect the semantics of PALPS need to be refined with the use of a new non-deterministic transition system. This new transition relation prunes away all process executions that do not respect the priority ordering defined by the applied policy. Precisely, given a PALPS system $S$ and a policy $\sigma$ then, the semantics of the initial configuration $(E, S)$ under the policy $\sigma$ is given by $\longrightarrow_p \cup \longrightarrow_\sigma$ where the prioritized nondeterministic transition relation $\longrightarrow_\sigma$ is defined by the following rule:

$$\frac{(E, S) \xrightarrow{\alpha}_n (E', S') \text{ and } (E, S) \not\xrightarrow{\beta}_n, (\alpha, \beta) \in \sigma}{(E, S) \xrightarrow{\alpha}_\sigma (E', S')}$$

### 2.4 Examples

*Example 1.* We consider a simplification of the model presented in [28] which studies the reproduction of the parasitic *Varroa mite*. This mite usually attacks honey bees and it has a pronounced impact on the beekeeping industry. In this system, a set of individuals reside on an $n \times n$ lattice of resource sites and go through phases of reproduction and dispersal. Specifically, the studied model considers a population where individuals disperse in space while competing for

a location site during their reproduction phase. They produce offspring only if they have exclusive use of a location. After reproduction the offspring disperse and continue indefinitely with the same behavior. In PALPS, we may model the described species $\mathbf{s}$ as $R \stackrel{\text{def}}{=} !rep.P_0$, where

$$P_0 \stackrel{\text{def}}{=} \sum_{\ell \in \mathbf{Nb}(\mathsf{myloc})} \frac{1}{|\mathbf{Nb}(\mathsf{myloc})|} : go\,\ell.\mathsf{cond}\ (\mathbf{s}@\mathsf{myloc} = 1 \rhd P_1;\ \mathsf{true} \rhd \surd.P_0)$$

$$P_1 \stackrel{\text{def}}{=} \overline{rep}.(p{:}\surd.P_0 \oplus (1-p){:}\overline{rep}.\surd.P_0)$$

We point out that the conditional construct allows us to determine the exclusive use of a location by an individual. The special label $\mathsf{myloc}$ is used to denote the actual location of an individual within a system definition. Furthermore, note that $P_1$ models the probabilistic production of one or two children of the species. During the dispersal phase, an individual moves to a neighboring location which is chosen equiprobably among the neighbors of its current location. A system that contains two individuals at a location $\ell$ and one at location $\ell'$ can be modeled as

$$System \stackrel{\text{def}}{=} (P_0{:}\langle \ell, \mathbf{s}, 2\rangle | P_0{:}\langle \ell', \mathbf{s}\rangle | (!rep.P_0){:}\langle \mathbf{s}\rangle) \backslash \{rep\}.$$

In order to refine the system so that during each cycle of the individuals' lifetime all dispersals take place before the reproductions, we may employ the policy $\{(\tau_{rep,\ell,\mathbf{s}}, \tau_{go,\ell',\mathbf{s}}) | \ell, \ell' \in \mathbf{Loc}\}$. Then, according to the PALPS semantics, possible executions of $System$ have the form:

$$\begin{aligned} System \ &\xrightarrow{w}_p\ (go\,\ell_1.\ldots{:}\langle \ell, \mathbf{s}\rangle | go\,\ell_2.\ldots{:}\langle \ell, \mathbf{s}\rangle | go\,\ell_3.\ldots{:}\langle \ell', \mathbf{s}\rangle) \backslash \{rep\} \\ &\xrightarrow{\tau_{go,\ell_1,\mathbf{s}}}_\sigma (\mathsf{cond}\ (\ldots){:}\langle \ell_1, \mathbf{s}\rangle | go\,\ell_2.\ldots{:}\langle \ell, \mathbf{s}\rangle | go\,\ell_3.\ldots{:}\langle \ell', \mathbf{s}\rangle) \backslash \{rep\} \end{aligned}$$

for some probability $w$ and locations $\ell_1, \ell_2, \ell_3$, where no component will be able to execute the $\overline{rep}$ action before all components finish executing their movement actions.

*Example 2.* Let us now extend the previous example into a two-species system. In particular, consider a competing species $\mathbf{s}'$ of the Varroa mite, such as the pseudo-scorpion, which preys on $\mathbf{s}$. To model this, we may define the process $R \stackrel{\text{def}}{=} !rep'.Q_0$, where

$$\begin{aligned} Q_0 &\stackrel{\text{def}}{=} \mathsf{cond}\ (s@\mathsf{myloc} \geq 1 \rhd Q_1, s@\mathsf{myloc} < 1 \rhd Q_2) \\ Q_1 &\stackrel{\text{def}}{=} \overline{prey}.Q_3 + \overline{rep'}.Q_4 \\ Q_2 &\stackrel{\text{def}}{=} \overline{rep'}.\surd.Q_5 \\ Q_3 &\stackrel{\text{def}}{=} \overline{rep'}.\surd.Q_0 \\ Q_4 &\stackrel{\text{def}}{=} \mathsf{cond}\ (s@\mathsf{myloc} \geq 1 \rhd \overline{prey}.\surd.Q, s@\mathsf{myloc} < 1 \rhd \surd.Q_5) \\ Q_5 &\stackrel{\text{def}}{=} \mathsf{cond}\ (s@\mathsf{myloc} \geq 1 \rhd \overline{prey}.Q_3, s@\mathsf{myloc} < 1 \rhd \mathbf{0}) \end{aligned}$$

An individual of species $s'$ initially has a choice between preying or producing an offspring. If it succeeds in locating a prey then it preys on it. If it fails then it makes another attempt in the next cycle. If it fails again then it dies.

To implement the possibility of preying on the side of **s**, its definition must be extended with complementary input actions on channel *prey* at the appropriate places:

$$P_0 \stackrel{\text{def}}{=} \sum_{\ell \in \mathbf{Nb}(\text{myloc})} \frac{1}{|\mathbf{Nb}(\text{myloc})|} : (go\,\ell.\mathsf{cond}\ (\mathbf{s}@\text{myloc} = 1 \rhd P_1;\ \mathsf{true} \rhd \surd.P_0) + prey.\mathbf{0})$$

$$P_1 \stackrel{\text{def}}{=} \overline{rep}.(p{:}\surd.P_0 \oplus (1-p){:}\overline{rep}.\surd.P_0) + prey.\mathbf{0}$$

In this model it is possible to define an ordering between the actions of a single species, between the actions of two different species or even between actions on which individuals of the two different species synchronize. For instance, to specify that preying takes place in each round before individuals of species **s** disperse and before individuals of species $\mathbf{s}'$ reproduce we would employ the policy

$$\sigma = \{(\tau_{go,\ell,\mathbf{s}}, \tau_{prey,\ell,\mathbf{s}'}), (\tau_{rep',\ell,\mathbf{s}'}, \tau_{prey,\ell,\mathbf{s}'}) | \ell \in \mathbf{Loc}\}.$$

Furthermore, to additionally require that reproduction of species **s** precedes reproduction of species $\mathbf{s}'$, we would write $\sigma \cup \{(\tau_{rep',\ell,\mathbf{s}'}, \tau_{rep,\ell,s}) | \ell \in \mathbf{Loc}\}$.

## 3 Translating PALPS into PRISM

In this section we turn to the problem of model checking PALPS models extended with policies. As is the case of PALPS without policies, the operational semantics of PALPS *with policies* gives rise to transition systems that can be easily translated to Markov decision processes (MDPs). As such, model checking approaches that have been developed for MDPs can also be applied to PALPS models. PRISM is one such tool developed for the analysis of probabilistic systems. Specifically, it is a probabilistic model checker for Markov decision processes, discrete time Markov chains, and continuous time Markov chains. For our study we are interested in the MDP support of the tool which offers model checking and simulation capabilities of PRISM models.

In [24] we defined a translation of PALPS into the MDP subset of the PRISM language and we explored the possibility of employing the probabilistic model checker PRISM to perform analysis of the semantic models derived from PALPS processes. In this paper, we refine the translation of [24] for taking policies into account. In the remainder of this section, we will give a brief presentation of the PRISM language, sketch an encoding of (a subset of) PALPS *with policies* into PRISM and state its correctness.

### 3.1 The PRISM Language

The PRISM language is a simple, state-based language, based on guarded commands. A PRISM model consists of a set of *modules* which can interact with each

other on shared actions following the CSP-style of communication [1]. Each module possesses a set of *local variables* which can be written by the module and read by all modules. In addition, there are *global variables* which can be read and written by all modules. The behavior of a module is described by a set of *guarded commands*. When modeling Markov decision processes, these commands take the form:

`[act] guard` $p_1$ `:` $u_1$ `+ ... +` $p_m$ `:`$u_m$`;`

where `act` is an optional action label, `guard` is a predicate over the set of variables, $p_i \in (0, 1]$ and $u_i$ are updates of the form:

`(`$x'_1$ `=` $u_{i,1}$`) & ... & (`$x'_k$ `=` $u_{i,k}$`)`

where $u_{i,j}$ is a function over the variables. Intuitively, such an action is enabled in global state $s$ if $s$ satisfies `guard`. If a command is enabled then it may be executed in which case, with probability $p_i$, the update $u_i$ is performed by setting the value of each variable $x_j$ to $u_{i,j}(s)$ (where $x'_j$ denotes the new value of variable $x_j$).

A model is constructed as the parallel composition of a set of modules. The semantics of a complete PRISM model is the parallel composition of all modules using the standard CSP parallel composition. This means that all the modules synchronize over all their common actions (i.e., labels). For a transition arising from synchronization between multiple processes, the associated probability is obtained by multiplying those of each component transition. Whenever, there is a choice of more than one commands, this choice is resolved non-deterministically. We refer the reader to [1] for the full description and the semantics of the PRISM language.

### 3.2 Encoding PALPS *with Policies* into the PRISM Language

As observed in [19], the main challenge of translating a CCS-like language (like PALPS) into PRISM is to map binary CCS-style communication over channels to PRISM's multi-way (CSP-style) communication. Our approach for dealing with this challenge in [24], similarly to [19], was to introduce a distinct action for each possible binary, channel-based communication which captures the channel as well as the sender/receiver pair.

In PALPS *with policies* the translation becomes more complex because, at any point, we need to select actions that are not preempted by other enabled actions. For, suppose that a policy $\sigma$ specifies that $(\alpha, \beta) \in \sigma$. This implies that, at any point during computation, we must have information as to whether $\beta$ is enabled. To implement this in PRISM, we employ a variable $n_\beta$ which records the number of $\beta$s enabled. To begin with, this variable must be appropriately initialized. Subsequently, it is updated as computation proceeds: once a $\beta$ is executed then $n_\beta$ is decreased by 1 and when a new occurrence becomes enabled it is increased by 1. Thus, if $(\alpha, \beta) \in \sigma$, execution of action $\alpha$ in any module of a model should have as a precondition that $n_\beta = 0$.

To translate PALPS into the PRISM language, we translate each process into a module. The execution flow of a process is captured with the use of a local variable within the module whose value is updated in every command in such as way that computation is guided through the states of the process. Then, each possible construct of PALPS is modeled via a set of commands. For example, the probabilistic summation is represented by encoding the probabilistic choices into a PRISM guarded command. Non-deterministic choices are encoded by a set of simultaneously enabled guarded commands that capture all non-deterministic alternatives, whereas the conditional statement is modeled as a set of guarded commands, where the guard of each command is determined by the expressions of the conditional process.

Unfortunately, the replication operator cannot be directly encoded into PRISM since the PRISM language does not support the dynamic creation of modules. To overcome this problem, we consider a bounded-replication construct of the form $!^m P$ in which we specify the maximum number of $P$'s, $m$, that can be created during computation.

In the remainder of this section we present the translation of a simple PALPS model into PRISM considering the main ideas of the encoding. This model is an instantiation of the model in Example 1. The full details of the translation can be found in [23].

*Example 3.* Consider a habitat consisting of four patches $\{1, 2, 3, 4\}$, where **Nb** is the symmetric closure of the set $\{(1, 2), (1, 3), (2, 4), (3, 4)\}$. Let **s** be a species residing on this habitat defined according to the bounded replication $R \stackrel{\text{def}}{=} !^m rep.P_0$ and where:

$$P_0 \stackrel{\text{def}}{=} \sum_{\ell \in \mathbf{Nb}(\mathsf{myloc})} \frac{1}{2} : go\,\ell.\mathsf{cond}\ (\mathbf{s}@\mathsf{myloc} = 1 \rhd P_1;\ \mathsf{true} \rhd \surd.P_0)$$

$$P_1 \stackrel{\text{def}}{=} \overline{rep}.(0.7{:}\surd.P_0 \oplus 0.3{:}\overline{rep}.\surd.P_0)$$

Now, consider a system initially consisting of two individuals, at locations 1 and 2:

$$System \stackrel{\text{def}}{=} (P_0{:}\langle \mathbf{s}, 1\rangle \mid P_0{:}\langle \mathbf{s}, 2\rangle \mid R{:}\langle \mathbf{s}\rangle)\backslash\{rep\}$$

Further, suppose that we would like to analyze the system under the policy where dispersal precedes reproduction: $\{(\tau_{rep,\ell,\mathbf{s}}, \tau_{go,\ell',\mathbf{s}}) | \ell, \ell' \in \mathbf{Loc}\}$.

In order to translate *System* under policy $\sigma$ in the PRISM language we first need to encode global information relating to the system. This consists of four global variables that record the initial populations of each of the locations and a variable that records the number of enabled occurrences the higher-priority actions referred to in the policy $\sigma$, that is, of $\tau_{go,\mathbf{s},\ell}$. We also include a global variable $i$ that measures the inactivated individuals still available to be triggered. Initially $i = m$. Finally, we make use of a global variable *pact* which takes values from $\{0, 1\}$ and expresses whether there is a probabilistic action enabled. It is used to give precedence to probabilistic actions over nondeterministic actions as required by the PALPS semantics. Initially, $pact = 0$.

```
module P1

st1 : [1..12] init 1;
loc1: [1..4] init 1;

[prob] (st1=1) -> 0.5:(st1'=2)&(n_g'=n_g+1)&(pact'=0)
                  + 0.5:(st1'=3)&(n_g'=n_g+1)&(pact'=0);
[] (pact=0)&(st1=2)&(loc=1) -> (loc'=2)&(s1'=s1-1)&(s2'=s2+1)
                         &(n_g'=n_g-1)&(st'=4);
[] (pact=0)&(st1=3)&(loc=1) -> (loc'=3)&(s1'=s1-1)&(s3'=s3+1)
                         &(n_g'=n_g-1)&(st'=4);
... // All possible locations are enumerated

[] (pact=0)&(st1=4)&(loc=1)&(s1=1) -> (st1=5);
[] (pact=0)&(st1=4)&(loc=1)&(s1!=1) -> (st1=10);
... // All possible locations are enumerated

[] (pact=0)&(st1=5)&(i>0)&(n_g=0) -> (s1'=s1+1)&(i'=i-1)&(st'=6);
[rep_1_3] (pact=0)&(st1=6) -> (st1'=7)&(pact'=1);
... // All activation possibilities are enumerated
[prob] (pact=0)&(st1=7) -> 0.7:(st1'=10)&(pact'=0)
                           + 0.3:(st1'=8)&(pact'=0);
[] (pact=0)&(st1=8)&(i>0)&(n_g=0) -> (s1'=s1+1)&(i'=i-1)&(st'=9);
[rep_1_3] (pact=0)&(st1=9) -> (st1'=10);
... // All activation possibilities are enumerated

[tick] (st1=10) -> (st1'=11);
[] (st1=11) -> (pact' = 1)&(st1'=12);
[tick'] (st1=12) -> (st'=11);

[prob] (pact=1)&(st!=1)&(st!=7) -> (pact'=0)
endmodule
```

**Fig. 1.** PRISM code for an active individual

```
global s1, s2: [0,m+2] init 1;      global n_g: [0,m+2] init 0;
global s3, s4: [0,m+2] init 0;      global pact: [0,1] init 0;
global i: int init m;
```

We continue to model the two individuals $P_0{:}\langle \mathbf{s}, 1\rangle$ and $P_0{:}\langle \mathbf{s}, 2\rangle$. Each individual will be described by a module. In Fig. 1, we may see the translation of individual $P_0{:}\langle \mathbf{s}, 1\rangle$. Individual $P_0\langle \mathbf{s}, 2\rangle$ is defined similarly.

In the translation of $P_0{:}\langle \mathbf{s}, 1\rangle$, we observe that its location variable $loc1$ is set to 1 and variable $st1$, recording its state, is set to 1, the initial state of the module. Overall, the module has 12 different states. Furthermore, all non-probabilistic actions have $pact = 0$ as a precondition.

From state 1 the module may non-deterministically decide on the location to which the individual will disperse (horizontal or vertical dispersal) while variable

$n_g$ is increased by one as the *go* action becomes enabled. This is implemented from states 2 and 3 respectively where the number of individuals of the source and destination locations are updated accordingly and the variable $n_g$ is decreased by one as there is now one fewer movement action enabled. Then, in state 4 the module determines if there exist more than one individuals in its current location. If yes, then the state progresses to 10 where the individual will synchronize on the *tick*. Otherwise, in state 5, assuming that there is no dispersal action available for execution ($n_g = 0$) and there is still an inactive individual to be activated ($i > 0$), variables $i$ and $s_{loc}$ are updated and the flow of control is passed on to state 7 where a synchronization with an inactive module is performed. This process is repeated in states 7–9 where a second offspring may be produced probabilistically. Finally, we point out that the tick action is implemented via three actions in PRISM (states 10–12): initially all modules are required to synchronize on the *tick* action, then they all perform their necessary updates on variable *pact* since a probabilistic action has become enabled and, finally, the modules are required to synchronize again before they may start to execute their next time step.

Moving on to the encoding of $R$, as we have already discussed, we achieve this via bounded replication which makes an assumption on the maximum number of new individuals that can be created in a system. Given this assumption, our model must be extended by an appropriate number of inactive individuals awaiting for a trigger via a $rep\_i\_j$ action. It then proceeds with the code of $P_0$ just like an active individual.

Regarding the correctness of the proposed translation, we have proved the existence of a weak bisimulation between a PALPS model and its PRISM translation by showing that a move of the PALPS model can be mimicked by its translation in a finite number of steps, and that this set of steps is performed atomically, in the sense that no other action within the PRISM model may interfere with the execution of the PALPS step. Similarly, any move of the PRISM translation can be mimicked by the original PALPS system. This proof of correctness is presented in [23].

## 4 A Case Study in PRISM

In this section, we apply our methodology for the simulation and model checking of PALPS systems using the PRISM tool. We begin by considering the system in Example 1, Sect. 2.4, which was also considered in [24] and can thus serve as a benchmark for studying the effect of applying policies on systems and, in particular, the degree by which policies reduce the state space of a PRISM model. In our model we will assume a lattice of locations of size $n \times n$ (for $n = 4, 9, 16$). Furthermore, we assume periodic boundaries conditions so that the opposite sides of the grid are connected together and we instantiate $p = 0.4$.

The PRISM encoding of the system follows the translation methods presented in Sect. 3. We performed some obvious optimizations in order to reduce the size of our model. All the tests were performed on a G46VW Asus laptop with an

**Table 1.** Performance of building probabilistic models in PRISM with and without policies.

| Case study size | Number of states | Number of transitions | Construction time (s) | RAM (GB) |
|---|---|---|---|---|
| No policy [24] | | | | |
| 3 PALPS individuals | 130397 | 404734 | 8 | 0.5 |
| 4 PALPS individuals | 1830736 | 7312132 | 101 | 1.9 |
| Policy $\sigma$ | | | | |
| 3 PALPS individuals | 27977 | 64282 | 3 | 0.3 |
| 4 PALPS individuals | 148397 | 409342 | 10 | 0.7 |
| Extended policy | | | | |
| 3 PALPS individuals | 20201 | 41602 | 3 | 0.3 |
| 4 PALPS individuals | 128938 | 310393 | 9 | 0.6 |

Intel i5 2.50 GHz processor and 8 GB of RAM. We ran the tests under Linux Ubuntu 13.04 (Kernel 3.8.0_17), using PRISM 4.0.3 with the MTBDD engine for model checking and CI method for simulation, and Java 7.

As a first experiment, we explored and compared the effect of applying policies on the state space of the system in question. Specifically, individuals in the system may engage in two activities: reproduction and dispersal. Let us assume an ordering of these two activities so that reproduction follows dispersal. This gives rise to the policy $\sigma = \{(\tau_{rep,\ell,\mathbf{s}}, \tau_{go,\ell,\mathbf{s}}) \mid \ell \in \mathbf{Loc}\}$. In Table 1, we summarize the results we obtained. We may observe that applying policy $\sigma$ has resulted in a reduction in the size of the states spaces by a factor of 10 (see cases *No policy* and *Policy* $\sigma$). A further reduction was achieved by further extending our policy to enforce an order between the execution of actions among individuals. Specifically, for each action (e.g., reproduction), the individuals executed the action in an increasing order in terms of their module identifier. This extended policy resulted in a further reduction of the state space by about 20 %.

As a second experiment, we attempted to determine the limits for *simulating* PALPS models. We constructed PRISM models with various numbers of modules of active and inactive individuals and we run them on PRISM. In Table 2, we summarize the results. It turns out that for models with more than 5000 individuals simulation requires at least 12 h (which was the time limit we set for our simulations).

Consequently, we redeveloped our model of the *Varroa mite* according to the description presented in [28]. In contrast to Example 1, the new model features mortality. Specifically, the new model has two parameters: $b$ the *offspring size* and $p$ the *probability to survive before breeding*. Each mite begins its life by being exposed to death and it survives with a probability $p$. In case of survival, it disperses to a new location. If it has exclusive use of the location then it produces an offspring of size $b$ and it dies. If the location is shared with other mites then all mites die without reproducing. As before, we model space as a lattice with periodic boundary conditions and the probability of dispersal from a location to any of its four neighbors equal to 1/4. As in the previous example,

**Table 2.** Performance of simulating probabilistic systems in PRISM.

| Individuals | File size (MB) | RAM (GB) | Simulation time (s) |
|---|---|---|---|
| 10 | 0.1 | 0.18 | 1 |
| 100 | 0.4 | 0.3 | 8 |
| 500 | 2.0 | 0.5 | 45 |
| 1000 | 4.2 | 1.0 | 300 |
| 1500 | 6.2 | 0.7 | 454 |
| 2000 | 8.2 | 0.9 | 820 |
| 5000 | 20.1 | 2.0 | $> 12$ h |
| 10000 | 44.1 | 3.4 | $> 12$ h |

in our system we employed the policy specifying that the process of dispersal precedes reproduction. Formally, the behavior of a mite is defined as follows:

$$P \stackrel{\text{def}}{=} p{:}P_1 + (1-p){:}\surd.0$$

$$P_1 \stackrel{\text{def}}{=} \sum_{\ell \in \mathbf{Nb}(\mathsf{myloc})} \frac{1}{4} : go\,\ell.\mathsf{cond}\ (\mathbf{s}@\mathsf{myloc} = 1 \rhd P_2;\ \mathsf{true} \rhd \surd.0)$$

$$P_2 \stackrel{\text{def}}{=} \overline{rep}^b.\surd.0 \qquad \text{where } \overline{rep}^b \stackrel{\text{def}}{=} \underbrace{\overline{rep}...\overline{rep}}_{b \text{ times}}$$

For our experiments, we took advantage of the model checking capabilities of PRISM and we checked properties by using the *model-checking by simulation* option, referred to as *confidence interval* (CI) simulation method. The property we experimented with is $R =?[I = k]$. This property is a reward-based property that computes the average state instant reward at time $k$. We were interested to study the expected size of the population. For this, we associate to each state a reward representing this size. In our experiments, we varied the size of the initial population ($i$), while the probability of surviving ($p$) and the offspring size ($b$) were fixed to $p = 0.9$ and $b = 3$, and the lattice was of size $4 \times 4$. The number of idle processes was fixed to $n \times b - i$, which is sufficient to avoid deadlocks. The results of the experiments, shown in Fig. 2, demonstrate a tendency of convergence to a stable state and an independence of the initial population for $i > 8$.

We also analyzed, for this model, the effect of the parameters $b$ and $p$ on the evolution of the average total number of individuals through time, with an initial population of 1 individual, as shown in Figs. 3, 4. The chosen values for $p$ and $b$ were selected so that they are close to the estimates of the parameters of the Varroa mite, namely, $b = 3$ and $p = 0.9$. Finally, we note that the results may also be applicable to other species that follow the same, so-called scramble-contest, behavior such as the bean bruchid that attacks several kind of beans.

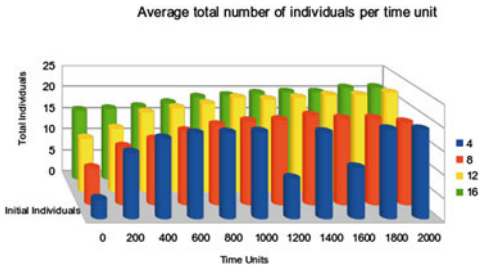


**Fig. 2.** Expected population size vs simulation time for different initial sizes of the population.

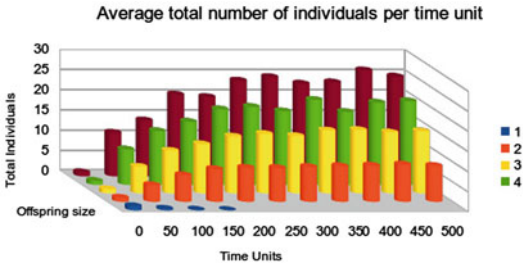


**Fig. 3.** Expected population size vs simulation time for different offspring sizes, for $p = 0.9$ and $i = 1$.

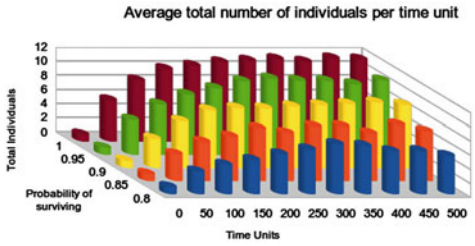


**Fig. 4.** Expected population size vs simulation time for different probabilities of survival, for $b = 3$ and $i = 1$.

## 5 Conclusions

In this paper we have extended the process calculus PALPS with the notion of a *policy*. A policy is an entity that is defined externally to the process-calculus description of an ecological system in order to impose an ordering between the activities taking place within a system as required for the purposes of the analysis. Furthermore, we have described a translation of PALPS *with policies* into the PRISM language. This encoding can be employed for simulating and model checking PALPS systems using the PRISM tool. We experimented with both of these

capabilities and we have illustrated types of analysis that can be performed on PALPS models. We have also contrasted our results with those obtained for the same example in our previous work [24]. We have concluded that applying policies can significantly reduce the size of the model thus allowing to consider larger models. For instance, in the example we considered, the state space of the model was reduced by a factor of 10.

As future work, we intend to investigate further approaches for analysis of MDPs that arise from the modeling of population systems. One such approach involves the PRISM tool and concerns the production of PRISM input: we intend to explore alternatives of producing such input possibly via constructing and providing PRISM directly the Markov decision process associated with a PALPS system. We expect that this will result in smaller state spaces than those arising via our PALPS-to-PRISM translation. Furthermore, we would like to explore other directions for reducing the state-space of PALPS models e.g. by defining an enhanced semantics of PALPS to enable a more succinct presentation of systems especially in terms of the multiplicity of individuals, as well as defining a symbolic semantics which applies a symbolic treatment of environments.

Another direction that we are currently exploring is the application of our methodology to new and complex case studies from the local habitat and the exploration of properties such as extinction (e.g., the expected time until extinction), persistence (e.g., the long-term average of the number of sites occupied at a given time) and spatial indices (e.g., the correlation among nearby locations in space, patch shape analysis and the number of subpopulations in a spatially dispersed metapopulation) similarly to [27].

Finally, an interesting future research direction would be extend the work of [16] towards the development of mean-field analysis to represent the average behavior of systems within a spatially-explicit framework.

## References

1. Online PRISM documentation. http://www.prismmodelchecker.org/doc/
2. Barbuti, R., Maggiolo-Schettini, A., Milazzo, P., Pardini, G.: Spatial calculus of looping sequences. Theoret. Comput. Sci. **412**(43), 5976–6001 (2011)
3. Barbuti, R., Maggiolo-Schettini, A., Milazzo, P., Troina, A.: A calculus of looping sequences for modelling microbiological systems. Fund. Inf. **72**(1–3), 21–35 (2006)
4. Berec, L.: Techniques of spatially-explicit individual-based models: construction, simulation, and mean-field analysis. Ecol. Model. **150**, 55–81 (2002)
5. Besozzi, D., Cazzaniga, P., Pescini, D., Mauri, G.: Modelling metapopulations with stochastic membrane systems. BioSystems **91**(3), 499–514 (2008)
6. Bioglio, L., Calcagno, C., Coppo, M., Damiani, F., Sciacca, E., Spinella, S., Troina, A.: A Spatial Calculus of Wrapped Compartments. CoRR, abs/1108.3426 (2011)
7. Cardona, M., Colomer, M.A., Margalida, A., Palau, A., Pérez-Hurtado, I., Pérez-Jiménez, M.J., Sanuy, D.: A computational modeling for real ecosystems based on P systems. Nat. Comput. **10**(1), 39–53 (2011)
8. Chen, Q., Ye, F., Li, W.: Cellular-automata-based ecological and ecohydraulics modelling. J. Hydroinf. **11**(3/4), 252–272 (2009)

9. Ciocchetta, F., Hillston, J.: Bio-PEPA: a framework for the modelling and analysis of biological systems. Theoret. Comput. Sci. **410**(33–34), 3065–3084 (2009)
10. Cleaveland, R., Lüttgen, G., Natarajan, V.: Priority in process algebras. Technical report, Langley Research Center, NASA, USA (1999)
11. Drábik, P., Maggiolo-Schettini, A., Milazzo, P.: Modular verification of interactive systems with an application to biology. Sci. Ann. Comp. Sci. **21**(1), 39–72 (2011)
12. Gerber, L.R., VanBlaricom, G.R.: Implications of three viability models for the conservation status of the western population of Steller sea lions (Eumetopias jubatus). Biol. Conserv. **102**, 261–269 (2001)
13. Hoare, C.A.R.: Communicating Sequential Processes. Prentice-Hall, Upper Saddle River (1985)
14. McCaig, C., Fenton, A., Graham, A., Shankland, C., Norman, R.: Using process algebra to develop predator–prey models of within-host parasite dynamics. J. Theor. Biol. **329**, 74–81 (2013)
15. McCaig, C., Norman, R., Shankland, C.: Process algebra models of population dynamics. In: Horimoto, K., Regensburger, G., Rosenkranz, M., Yoshida, H. (eds.) AB 2008. LNCS, vol. 5147, pp. 139–155. Springer, Heidelberg (2008)
16. McCaig, C., Norman, R., Shankland, C.: From individuals to populations: a mean field semantics for process algebra. Theoret. Comput. Sci. **412**(17), 1557–1580 (2011)
17. Milner, R.: A Calculus of Communicating Systems. Springer, Heidelberg (1980)
18. Milner, R., Parrow, J., Walker, D.: A calculus of mobile processes, parts 1 and 2. Inf. Comput. **100**, 1–77 (1992)
19. Norman, G., Palamidessi, C., Parker, D., Wu, P.: Model checking probabilistic and stochastic extensions of the $\pi$-calculus. IEEE Trans. Softw. Eng. **35**(2), 209–223 (2009)
20. Pardini, G.: Formal modelling and simulation of biological systems with spatiality. Ph.D thesis, University of Pisa (2011)
21. Pearson, R.G., Dawson, T.P.: Long-distance plant dispersal and habitat fragmentation: identifying conservation targets for spatial landscape planning under climate change. Biol. Conserv. **123**, 389–401 (2005)
22. Pescini, D., Besozzi, D., Mauri, G., Zandron, C.: Dynamical probabilistic P-systems. J. Found. Comput. Sci. **17**(1), 183–204 (2006)
23. Philippou, A., Toro, M.: Process ordering in a process calculus for spatially-explicit ecological models. Technical report, Department of Computer Science, University of Cyprus, 2013. http://www.cs.ucy.ac.cy/~annap/pt-tr.pdf
24. Philippou, A., Toro, M., Antonaki, M.: Simulation and verification for a process calculus for spatially-explicit ecological models. Sci. Ann. Comput. Sci. **23**(1), 119–167 (2013)
25. Păun, G.: Computing with membranes (P systems): an introduction. In: Rozenberg, G., Salomaa, A. (eds.) Current Trends in Theoretical Computer Science, pp. 845–866. World Scientific, Singapore (2001)
26. Regev, A., Panina, E.M., Silverman, W., Cardelli, L., Shapiro, E.: BioAmbients: an abstraction for biological compartments. Theoret. Comput. Sci. **325**(1), 141–167 (2004)
27. Ruxton, G.D., Saravia, L.A.: The need for biological realism in the updating of cellular automata models. Ecol. Model. **107**, 105–112 (1998)
28. Sumpter, D.J.T., Broomhead, D.S.: Relating individual behaviour to population dynamics. Proc. Roy. Soc. B: Biol. Sci. **268**(1470), 925–932 (2001)
29. Tofts, C.: Processes with probabilities, priority and time. Formal Aspects Comput. **6**, 536–564 (1994)