

Population Studies

A Journal of Demography

ISSN: 0032-4728 (Print) 1477-4747 (Online) Journal homepage: www.tandfonline.com/journals/rpst20

Modelling and simulating decision processes of linked lives: An approach based on concurrent processes and stochastic race

Tom Warnke, Oliver Reinhardt, Anna Klabunde, Frans Willekens & Adelinde M. Uhrmacher

To cite this article: Tom Warnke, Oliver Reinhardt, Anna Klabunde, Frans Willekens & Adelinde M. Uhrmacher (2017) Modelling and simulating decision processes of linked lives: An approach based on concurrent processes and stochastic race, Population Studies, 71:sup1, 69-83, DOI: [10.1080/00324728.2017.1380960](https://doi.org/10.1080/00324728.2017.1380960)

To link to this article: <https://doi.org/10.1080/00324728.2017.1380960>



© 2017 The Author(s). Published by Informa UK Limited, trading as Taylor & Francis Group



[View supplementary material](#)



Published online: 24 Oct 2017.



[Submit your article to this journal](#)



Article views: 1760



[View related articles](#)



[View Crossmark data](#)



Citing articles: 6 [View citing articles](#)

Modelling and simulating decision processes of linked lives: An approach based on concurrent processes and stochastic race

Tom Warnke ¹, Oliver Reinhardt¹, Anna Klabunde ², Frans Willekens ²
and Adelinde M. Uhrmacher ¹

¹University of Rostock, ²Netherlands Interdisciplinary Demographic Institute (NIDI)

Individuals' decision processes play a central role in understanding modern migration phenomena and other demographic processes. Their integration into agent-based computational demography depends largely on suitable support by a modelling language. We are developing the Modelling Language for Linked Lives (ML3) to describe the diverse decision processes of linked lives succinctly in continuous time. The context of individuals is modelled by networks the individual is part of, such as family ties and other social networks. Central concepts, such as behaviour conditional on agent attributes, age-dependent behaviour, and stochastic waiting times, are tightly integrated in the language. Thereby, alternative decisions are modelled by concurrent processes that compete by stochastic race. Using a migration model, we demonstrate how this allows for compact description of complex decisions, here based on the Theory of Planned Behaviour. We describe the challenges for the simulation algorithm posed by stochastic race between multiple concurrent complex decisions.

Supplementary material for this article is available at: <http://dx.doi.org/10.1080/00324728.2017.1380960>

Keywords: simulation models; theoretical models; migration

Background

Although agent-based models are increasingly popular in computational demography, tools that support scientists in designing and using such models remain scarce. In other domains that rely heavily on simulation studies, such as cell biology, a plethora of supporting tools has evolved, such as domain-specific modelling languages with corresponding simulation algorithms. As an alternative to general-purpose approaches, domain-specific languages aim to provide a more succinct, unambiguous, and readable description of systems of a certain domain. Metaphors from the application domain can be tightly integrated into the language and ease access to the model for domain experts. Many languages developed for cell biology, for example, share the ability to assign attributes to the entities and to constrain biochemical reactions based on these attributes, whether they describe the system of interest as attributed concurrent processes (John et al. 2010), as coloured Petri Nets (Pârvu et al.

2015), or as rules on attributed, hierarchically nested species (John et al. 2011; Maus et al. 2011).

These success stories motivated us to aim to identify specific features that should likewise be central in designing domain-specific modelling languages for computational demography. The Modelling Language for Linked Lives (ML3) (Warnke, Klabunde et al. 2015) is a domain-specific language that we are in the process of designing for agent-based computational demography. It focuses on agents, their links, behaviour, and interactions. In the following, we will analyse how complex decision processes in continuous time can be modelled, and what features may be essential in a domain-specific modelling language to do so comfortably.

Currently, agent-based demographic models are often implemented using generic frameworks for agent-based simulation. In such tools the model is described completely in a high-level programming language, for example, Java or C/C++ (Kravari and Bassiliades 2015). Length and complexity of the code hamper the inclusion of the model in

publications, and thus their validation and reuse (Grimm et al. 2010). NetLogo is considered an exception as far as ease of use is concerned, as it was designed explicitly for modellers without a programming background (Wilensky 1999). However, as with most tools that support agent-based simulation (Theodoropoulos et al. 2009), NetLogo executes models in a time-stepped manner. Only recently was a NetLogo extension developed to schedule events in continuous time (Sheppard and Railsback 2015), thereby reflecting requirements to capture the temporal behaviour of systems more realistically. However, the extension does not support the retraction of scheduled events, making it cumbersome for usage in demography, particularly when decisions frequently interrupt and change events that are already scheduled. This forces the modeller to integrate elaborate event cancelling strategies manually as part of the model (Klabunde et al. 2015).

Decision processes have been modelled in a variety of ways in demographic simulation models. Traditional demographic microsimulation models recreate the life course as a sequence of states that the individual occupies. Usually, the waiting time before transition to a new state is stochastic. If several successor states are possible, the state with the shortest waiting time drawn is chosen. This method to resolve nondeterminism is called ‘stochastic race’. The underlying assumption is that the more attractive an option is, the higher the probability of observing it within a certain time interval will be. Thus, a more attractive option corresponds to a shorter average waiting time and is more likely to be chosen.

Sometimes distributions from which waiting times are drawn are purely empirical, without reference to any causal underlying mechanism. Whereas this might be a reasonable approximation in many cases, for other decisions relevant to demography, decision theories from psychology, such as the Theory of Planned Behaviour (TPB), have been suggested for the decision to have a child (Ajzen and Klobas 2013) or for decisions relating to health (Godin and Kok 1996). Discrete choice models, which are popular in economics, can be transformed into, or at least be approximated by, a logit model (Train 2009), whereby a random draw from a uniform distribution then determines the choice. The TPB and utility theory have both been used extensively in agent-based models of demographic content; for an overview, see Klabunde and Willekens 2016. Heuristic decision rules based on social influence are also very popular when modelling fertility decisions and especially mating decisions (Billari et al. 2007; Fent et al. 2013), where search theory

plays an important role (Todd et al. 2005; Hills and Todd 2008). Decisions are not only made at the individual level, but also at the household level (e.g., Walsh et al. 2013), thus requiring that the different organizational levels are suitably reflected in the modelling language.

Thus, a modelling language for demographic simulation should allow for rate-based transitions whose sojourn times are drawn from purely empirical distributions. In addition, transitions that result from complex decision processes and depend, deterministically or stochastically, on the current context of an individual need to be supported in a flexible and convenient way.

ML3—language concepts

Before developing ML3, we evaluated the usefulness of existing modelling languages for agent-based computational demography (Steiniger et al. 2014). We found that some specifics of this domain, such as the explicit use of time to make statements about the age of agents, are not sufficiently supported by existing modelling approaches. In addition, the metaphors used (e.g., biochemical reactions) were not relevant, the solutions did not support a compact description of systems of linked individuals, or the syntax was unnecessarily complex. However, among the different approaches analysed, a process algebra allowed the most compact description of the example. Therefore, we based ML3 on concurrent processes, the underlying metaphor of process algebras. This allows for a natural representation of agents interacting in continuous time (see Feng and Hillston 2014; Zunino et al. 2015). We also decided to use a text-based notation of the model, which has advantages over graphical notations, such as better use of screen space, faster editing, and tool independence, at least for larger models (Petre 1995; Grönninger et al. 2007).

Thus, a model description in ML3 is contained in a text file. The contents of such a model file are structured into four consecutive sections—declarations of agent types, links, functions, and behaviour rules—as described next.

Agents and attributes

Agents represent all acting entities of the system to be modelled. We explicitly do not restrict agents only to persons; they can also represent higher-level demographic actors, such as families or households.

This is in line with developments in agent-based modelling in demography (see Walsh et al. 2013). Each agent in the model has a type that determines its properties and behaviours.

The agent types of a model are specified by declarations of their name and typed attributes in the first section of the model description. The following ML3 snippet declares an agent type called `Person` with three attributes: `sex`, `weekly income`, and `capital`. For the attribute `sex`, the possible attribute values `f` and `m` are enumerated, whereas `weekly income` and `capital` are defined to be real numbers. Other available attribute types include Boolean values and integers. Default values are denoted by `:=`.

```
Person(sex:{ "f", "m" } ,
       weeklyIncome:real := 0,
       capital:real := 0)
```

For each agent type, an attribute that holds the age of an agent is defined implicitly. This attribute is managed automatically during simulations, for example, when an agent is created or ceases to exist. For agents that represent individuals, this intuitively corresponds to birth and death. For higher-level agents, such as families, this attribute can be interpreted as the time when a family is formed by marriage or the time when it is ended by divorce or the death of the last family member. Currently, agents that have ‘died’ are not removed from the model. Living agents may be influenced by dead ones, for example, persons may be influenced by past marriages. In a future version of ML3, dead agents will be completely removed if no living agent is influenced by them.

Links between agents

Agents are connected to social networks by links. As links among persons, links between persons and higher-level agents, and links among higher-level agents are all central to agent-based demography, we regard links to be on an equal footing with agents. This reflects the recent tendency to put more emphasis on the connections between demographic individuals, to obtain richer, more realistic models (Noble et al. 2012). Thus, the declaration of links between agents makes up the second section of a model description. The example in the next snippet defines a link with `Person` agents on each side.

```
parents:Person[ 2]  <->  [ 0-] Person:
children;
```

This link declaration states that each `Person` agent is connected to exactly two other `Person` agents, who take the role `parents` in this link. Conversely, the declared link also connects each `Person` agent to an arbitrary number of other `Person` agents, who take the role `children`. Both relations are declared together. This ensures, that if a `Person` agent, `A`, is one of the two parents of another `Person` agent, `B`, then `B` is also one of the children of `A`.

The next snippet shows how the values of attributes and links of a specific agent can be accessed by using the point operator, like in object-oriented programming languages (double slashes `//` denote code comments).

```
// let p be an agent of type Person
p.weeklyIncome // the income of p
p.parents // the parents of p
```

Functions and procedures

Agent-based demographic models contain increasingly complex phenomena. For example, in the previous section we introduced ML3’s support for many-to-many links between agents. A useful modelling language must allow for powerful operation on such structures while maintaining legibility. To encapsulate complex computations under a meaningful name, functions and procedures were introduced as the third section of a ML3 model description. While functions represent analytic operations that result in a value, procedures encapsulate actions that change the model state. Both extend the expressive power of ML3 but, more importantly, provide valuable syntactical shortcuts, leading to more readable and succinct model definitions.

Like methods in object-oriented programming, each function or procedure definition is bound to a specific agent type. The function or procedure can then be invoked for each agent of that type. The modeller can define a function by declaring its name, then its arguments with types (if any), followed by `:=` and its body. Procedures are defined in a similar way, with `->` instead of `:=`. In the body, the keyword `ego` refers to the specific agent that the function or procedure is invoked on, as shown here:

```
// a function
```

```

Person.isWealthy() := ego.capital >
  1.000.000;
// a procedure with a real-valued
argument termed "?amount"
Person.getRaise(?amount: real) ->
  ego.weeklyIncome += ?amount;

```

The body of a procedure can contain several statements that will be executed sequentially. Such an imperative style (i.e., stating how to achieve a result) rather than a declarative style (i.e., stating what result to achieve) is used in modelling languages in different domains, for example, the programme *ℓ* in systems biology (Zunino et al. 2015). It is also closely connected to programming languages, such as Java and C++, as well as R (R Core Team 2015). With R being a standard tool in demography, imperative procedures should be a familiar concept for many demographers.

Like the predefined attributes described in the previous subsection, for each agent there is a predefined function *isAlive()* (to check if an agent is alive) and a procedure *die()* (to end the life of an agent). Additionally, some functions are predefined to operate on sets, for example, sets obtained through links. Attributes and links, as well as function invocations, can be chained to create more complex expressions, such as

```

Person.isOrphan() :=
  ego.parents.filter(alter.isAlive
   ()).isEmpty();

```

The keyword *alter* refers to other agents than the one the function is invoked on, for example, agents connected to them by links. In the listing above, the function *filter* is invoked on the parents of *ego*. Thus, *alter* represents each one of the parents. To check whether the agent has no living parents, *isEmpty()* is invoked on the result of the filter function. This notion of set functions is again related to R and its higher-order functions.

Set functions also provide a means for agents to access global information. The keyword *all* can be used to refer to all agents of a type, for example, to calculate the average value of an attribute in the agent population:

```

// compute the average capital of all
  Person agents
Person.all.sum(alter.capital) /
  Person.all.size

```

Such expressions enable the implementation of agent behaviour dependent on macro-level information, that is, downwards causation.

Behavioural rules and waiting times

Rules describe the behaviour of agents and constitute the fourth and final section of a ML3 model. Like functions, rules are defined for an agent type. Besides the agent type, a rule definition consists of three components: conditions, a waiting time expression, and effects. The idea of this type of rule formulation is closely connected to stochastic guarded commands (Dijkstra 1975; Henzinger et al. 2011). This extract models the death of male agents of the type *Person*:

```

Person
| ego.sex = "m" // condition
@ maleDeathRate[ ego.age] // waiting
time expression
-> ego.die(); // effect

```

The conditions control which agents of this type are exposed to the rule. These conditions can make use of the attributes and links of the agent type. For each agent that is alive and satisfies all conditions, the rule is effective. By using functions, complex conditions can be defined while maintaining legibility.

The waiting time expression describes the waiting time until the rule is executed. This expression can encode a stochastic distribution of waiting times or a deterministic waiting time. ML3 supports different types of waiting time definition. If no keyword is given, the waiting time expression is interpreted as the rate of a (potentially non-homogeneous) Poisson process. Keywords such as *age* or *every* can be used to model agent behaviour on reaching a certain age or to repeat with a certain frequency. Waiting time distributions that take data into account, for example, age-specific rates, can be integrated easily. Again, attributes, links, and functions can be used to calculate agent-specific waiting times.

Finally, the effect of a rule describes the changes that are triggered by the execution of the rule. The changes are given as a list of imperative commands, such as assignments or procedure invocations. Examples are the creation and decease of agents or changes in attributes and links.

The next extract defines two rules for agents of the type *Person*. The first rule applies only to agents that are not wealthy (the symbol *!* negates conditions). It models winning a lottery. As no

keyword is given, its waiting time is exponentially distributed with the parameter `lotteryWinRate`, which is a model parameter. If its value is set to 1/100,000,000, the average waiting time for each agent is 100,000,000 time units.

```
Person
| !ego.isWealthy()
@ lotteryWinRate
-> ego.capital += 1.000.000;

@ every 7 synchronized
-> ego.capital += ego.weeklyIncome;
```

The second rule has no conditions and represents the weekly payment of wages. Its waiting time expression starting with `every` specifies a deterministic repeated rule execution every seven time units. The mapping of model time units to modelled time is up to the modeller; in this snippet, one time unit corresponds to one day. By adding the keyword `synchronized`, the modeller denotes that all `Person` agents will receive their weekly income simultaneously.

For each agent, multiple rules can be in effect simultaneously. This seamlessly integrates diverse agent behaviours with stochastic waiting times or deterministic timing. Thus, otherwise complex decision structures can be broken down to a set of parallel rules; this supports the development of readable and succinct models in ML3.

Simulating ML3 models

Simulation algorithms for models of competing transitions with exponentially distributed waiting times are well established, for example, the Doob–Gillespie algorithm (Doob 1945; Gillespie 1976). Given the state of the system at a certain time, these algorithms generate the next state and the time at which the system enters this next state. The distribution of this waiting time is defined by the stochastic rates of all transitions, usually dependent on the current state. Essentially these simulation algorithms determine the next time and state by sampling the waiting time distribution of all transitions and executing the transition with the shortest waiting time. The waiting time of a single transition is exponentially distributed with a rate only depending on the current state:

$$P(T \leq \Delta t) = 1 - \exp(-\lambda_s \Delta t)$$

where T denotes time (random variable), Δt denotes the time interval, and λ_s is the transition rate in the

current state s . This distribution can be easily sampled by generating a random number, r , from the uniform distribution on the unit interval and applying the distribution function's inverse:

$$\Delta t = \frac{1}{\lambda_s} \cdot \ln \frac{1}{r}.$$

ML3 also allows transition rates to be dependent on the age of agents or the current time. Thus, ML3 models are non-homogeneous continuous-time Markov chains. This results in a different waiting time distribution, which depends on the current time, t , and takes the change of transition rate $\lambda_s(t)$ during the waiting time into consideration (Jansen 1995):

$$P(T \leq \Delta t) = 1 - \exp\left(-\int_t^{t+\Delta t} \lambda_s(\tau) d\tau\right).$$

To draw from this distribution using the same method as noted previously, one must solve the equation

$$\int_t^{t+\Delta t} \lambda_s(\tau) d\tau = \ln \frac{1}{r}$$

for Δt . For the majority of transition rate functions, $\lambda_s(t)$, this cannot be done in an obvious efficient way. Numerical solutions are not feasible, as the equation must be solved for every transition just to generate one next state. For some rate functions, such as transition rates that do not change over time or transition rates that are piecewise constant, a closed form solution of the integral allowing the efficient solution of the equation can be obtained. As a consequence, ML3 only allows piecewise constant transition rates. Because demographic input data for rates are often age-specific and thus piecewise constant, this is not a strong restriction. Rate functions of other forms may be approximated, for example, as piecewise constant.

Note that this waiting time distribution is not memoryless, unlike the exponential distribution. The timing of events is generally not sampled from the same distribution after a state change. As time advances at each state change, the limits of the integral in the time-dependent waiting time distribution change. However, the value of the integral is still independent of the amount of time already spent in the state. Although this is a slightly weaker property than being memoryless, it suffices for Gillespie's algorithms.

In addition to transitions with stochastic waiting times, ML3 allows the scheduling of deterministic transitions by defining a time point of execution. The next executed transition will be the stochastic one with the shortest waiting time or the next

deterministically scheduled transition, whichever comes first.

Our first prototype of the simulation algorithm obtains new waiting times for all effective rules (deterministic and stochastic) after each event execution and executes only the fastest event (Figure 1). By determining which scheduled events are not affected by an event execution, we can avoid rescheduling all events. A first, improved, and more efficient simulation algorithm for ML3 has already been developed (Reinhardt and Uhrmacher 2017).

A model of migration decisions

As an illustration of the capabilities of ML3, we now show how the decision to migrate was modelled and implemented in an agent-based model of Senegalese people deciding whether or not to migrate to Europe. The model in itself is novel, due to its combination of a continuous-time competing risk framework and an established theory of decision-making. It is described in detail in Klabunde et al. (2015), hence we focus on the basics here. One purpose of the model is to offer an explanation for the age profile of migration. Another one is to provide a test bed for measuring the impact of policy change or the change of social norms on migration behaviour. The principal idea is that during a critical phase in life—around ages 18–40—migration ‘competes’ with other life decisions, such as getting married and having children.

In our model, there are two ‘countries’, representing Senegal and France. Our baseline model runs from 1980 to 2050, whereas we use available data from 1980 to 2011 for calibration. After 2011 we simply assume time series and parameter values as constant. In the model, Senegal is populated by individuals of all ages, taken directly from the 1982 Census population. They are distinguished by age, marital status, and (in the case of women) children ever born. When individuals receive their first wage, it is randomly drawn from a log-normal distribution constructed using the gross domestic product (GDP) per capita for Senegal and France in the respective year and the Gini coefficients of both countries. Wages are then updated every year according to the real growth in GDP in Senegal and France. Each family resides together in a household that pools incomes and distributes them evenly over members for consumption. Individuals belong to a network and are connected to their own family and to neighbours on a two-dimensional grid. During their life course, individuals are at risk of experiencing several demographic events: marriage,

childbirth, and death. The transition rates to these events are estimated empirically with the help of the MAFE (Migrations between Africa and Europe) data set (<http://mafeproject.site.ined.fr/en/>) and the Demographic and Health Survey for Senegal (<http://dhsprogram.com/>).

Furthermore, in the model, people in Senegal deliberate whether or not to migrate. For modelling the decision to migrate, we make heavy use of the TPB (Ajzen 1991). The general idea of this widely used and often tested theory of decision-making is that intentions determine behaviour (for a recent example, see Philipov et al. 2015). The choice of the TPB as the modelling framework is taken from the original model implementation in Klabunde et al. 2017 (in this Supplement) and is further described there.

Intentions are formed after taking into consideration one’s *attitude* towards a behaviour, *social norms* (e.g., the opinion of others), and one’s perceived ability to execute the behaviour (*perceived behavioural control*).

In our model, individuals have two possible motives to migrate: higher income and reuniting with family. The migration attitude, MA , of agent i at time t is therefore defined as

$$MA_{i,t} = ey_{i,t}py_{i,t} + ef_{i,t}pf_{i,t}$$

(see Klabunde et al. 2015), where ey is the evaluation of a higher income in the host country (how much the agent cares about a higher income); py is the subjective probability of achieving a higher income (how likely the agent is to actually achieve a higher income in the host country); ef is the evaluation of family reunification; and pf is the subjective probability of achieving family reunification (here assumed to be one). It would be straightforward to add in other migration motives by multiplying the evaluation of a possible migration outcome with the subjective belief of actually achieving this positive outcome.

The components of attitude formation are computed as follows. The evaluation of higher income in the host country is higher, the lower the household per capita wealth of the person is:

$$ey_{i,t} = a - \xi \frac{c_{h,i,t}}{A_{h,i,t}} \quad (1)$$

where $c_{h,i,t}$ is the capital of the household, h , that agent i belongs to at time t ; $A_{h,i,t}$ is the number of adults in the household, h , of agent i at time t ; and a and ξ are weighting parameters.

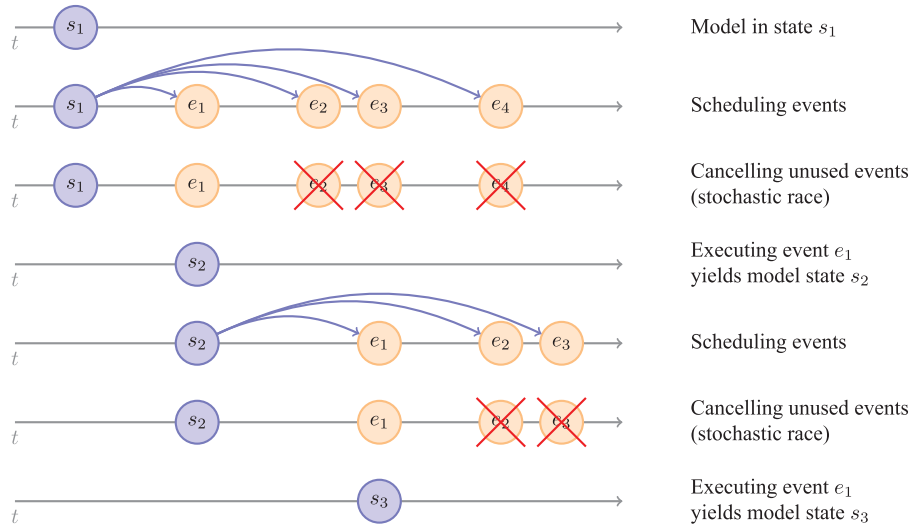


Figure 1 Example transition event generation and execution process in ML3 with several steps (from top to bottom). The current model state is denoted in blue^(s). Orange^(e) events are scheduled (stochastically or deterministically) and subject to stochastic race

Next,

$$ef_{i,t} = \zeta \# fam_{i,t}$$

where $\# fam_{i,t}$ is the number of previous family members who have migrated and ζ is a weighting parameter.

The subjective probability of achieving higher income by migrating, $py_{i,t}$, is computed as follows, based on past values the agent can observe. For each of the agent's network neighbours in the host country who is older than 16 years, the income after migration is compared with the agent's own income since they turned 16. The subjective probability, $py_{i,t}$, is then the proportion of the agent's network neighbours in the host country who have received higher incomes than the agent's income. The total number of person-day observations of network neighbours in the host country and the number of those observations which represented higher incomes are recorded, so that $py_{i,t}$ can be updated on any given day. If no migrants have been observed yet, we assume $py_{i,t} = 0.5$. That probability is low because, in the absence of observations, doubt is assumed to prevail. The subjective probability of family reunification is much higher because doubt is assumed to be considerably less.

The next factor determining intentions is what Ajzen calls 'social norms'. In our case, this factor is measured simply as the proportion of an individual's network neighbours who have ever migrated.

Finally, agents evaluate whether or not they consider it likely that they will be able to migrate successfully (perceived behavioural control). The two

factors that influence perceived behavioural control are their ability to afford the migration cost and the strictness of border enforcement. The perceived behavioural control of agent i at time t is

$$PBC_{i,t} = -(pb_{i,t}bc_t + pc_{i,t}fc_{i,t})$$

where bc_t is the externally enforced border control, which is the same for all individuals, and $pb_{i,t}$ describes (as before) for each individual the subjective probability that they will fail, that is, that the border control will be an effective deterrent. This probability is determined by the proportion of failed migration attempts that a person knows about. The migration costs are denoted by $fc_{i,t}$, and $pc_{i,t}$ is the subjective probability of the migration cost precluding the agent's migration.

The level of border enforcement, bc_t , is exogenous and assumed to be known to the agents, for example, through the media. Its level is updated three times between 1980 and 2011, reflecting important policy changes. The subjective probability that the border enforcement will hinder the agent's migration, $pb_{i,t}$, is the proportion of failed migration attempts among the agent's network neighbours. The migration cost per person is assumed to be fixed and to be lower if the agent knows someone who has migrated successfully. If the head of a one-parent household with children is considering migration, they would have to take their children to the host country, which would increase their total migration cost. The last element to be defined is the subjective probability of not being able to afford the migration cost, $pc_{i,t}$. The total quantity of funds that an agent has available to

pay the migration cost is the household's capital. The agent keeps track of their household capital, $c_{h,t}$ (see also equation (1)), and computes the average monthly capital of their current household. The proportion of months in which $c_{h,t} < fc_{i,t}$, that is, the proportion of months in which the agent would not have been able to afford the migration cost, determines the subjective probability of an agent not being able to meet the costs of migration, $pc_{i,t}$, at the time when they decide to migrate.

Together, migration attitude, social norms, and perceived behavioural control determine the intention to migrate. We assume a simple additive relationship:

$$I_{i,t} = \alpha MA_{i,t} + \beta SN_{i,t} + \gamma PBC_{i,t} \quad (2)$$

where α , β , and γ are weighting parameters. While attitude and social norms are positive values, PBC is negative, so the agent's intention, $I_{i,t}$, can be positive or negative.

So far, we have followed Ajzen's theory at face value. However, we must take into account that, usually, time passes between the first time a person considers the idea that migration might be an interesting option and the actual migration attempt. During that time, other life events can get in the way of migration and may postpone it, bring it forward, or supersede it. Thus, we transform the TPB into a process theory of decision-making (for further details on this novel approach, see Klabunde et al. 2017).

Individuals compute their migration intention for the first time when they reach age 17 and then after every demographic event. As long as an agent's intention is positive, they go through a planning and a preparation phase before eventually migrating. When an agent's intention becomes negative, they give up the idea of migrating. The waiting time in each stage, which is assumed to be exponentially distributed, is determined by the strength of the agent's intention. The decision process is depicted in Figure 2.

The transition rates, $\lambda_{i, \sigma_m \rightarrow \sigma_n}(t)$, of agent i passing at time t from one stage, m , to the next stage, n (that is, from intention to planning, from planning to preparation, and from preparation to migration), are defined as follows:

$$\lambda_{i, \sigma_m \rightarrow \sigma_n}(t) = \rho \exp(\theta I_{i,t}) \quad (3)$$

where ρ is the baseline rate for an intention value of 0.25; $I_{i,t}$ is the intention computed as in equation (2); and θ is a weighting parameter.

Entering a new stage in the migration decision process and experiencing a demographic event are competing risks. If a demographic event happens to an individual before the waiting time to the next

decision stage is over, a new intention value and a new waiting time for the next step in the migration decision are computed straight away, as shown in equation (3). This is necessary because the demographic event can change the evaluation of the attractiveness and feasibility of migration dramatically.

Once the waiting time in the final preparation stage is over, we make use of the TPB again: actual behavioural control, exogenous to the agent, determines if the intention is transformed into action. An agent's probability of successfully migrating, $PM_{i,t}$, is then determined based on the border control, bc_t , as

$$PM_{i,t} = \frac{1}{1 + \exp(-(1/bc_t))}, \quad bc_t > 0.$$

Modelling the migration decision with ML3

In this section, we discuss some snippets of the migration model expressed in ML3. We focus on the decision processes detailed in the previous section, and illustrate the design choices of our modelling approach based on this example. This will serve as an introduction to the complete model definition provided in the supplementary material.

Modelling migrating individuals

We begin with the definition of an agent type `Person`. This agent type represents the individual who decides whether to migrate or not. Among such an agent's attributes are the current stage in the decision process, the number of months in which they could have afforded migration, and the number of migration attempts, both overall and failed. Default values are given for most attributes, as shown in the next snippet. For example, newly created `Person` agents are children and thus, migration is not viable for them.

```
Person (
  sex: { "m", "f" },
  income: real := 0,
  mortalityModifier: real := 1,
  migrationStage: { "not viable",
    "intention", "planning",
    "preparation", "migrated",
    "exit" } := "not viable",
  migrationAttempts: int := 0,
  failedMigrationAttempts: int := 0,
  status: { "child", "adult",
    "retired" } := "child",
  canAffordMonths: int := 0,
```

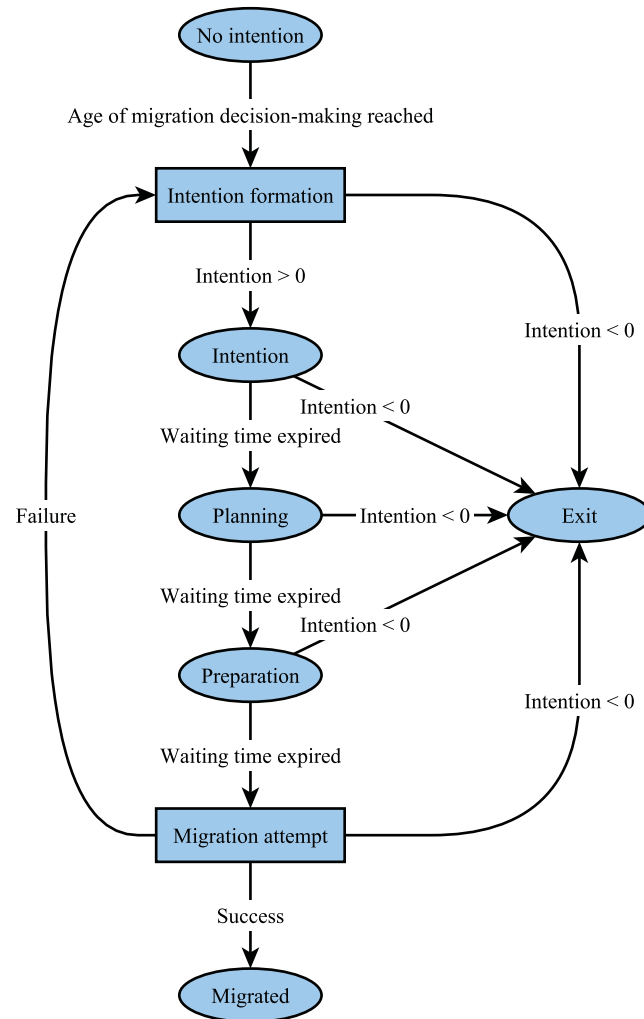


Figure 2 Possible transitions between different states in the migration decision model

```

canNotAffordMonths: int := 0,
migrationAge: real := 0,
migrationStartAge: real := 0
);

```

Modelling the spatial environment

One important requisite is to distinguish between individuals who are still at home and those who have already migrated to the host country. This is presented by an agent of type *Address* with a location attribute with corresponding values.

```

Address (
  location: { "home country," "host country" }
);

```

Relating an agent of type *Person* to a specific agent of type *Address* is done via links. Each

Person lives at exactly one *Address* at one point in time, whereas each *Address* can be inhabited by zero or more *Person* agents. Each *Address* is related to eight neighbouring *Address* agents. As the original model has been developed in NetLogo with its 2D environment, by linking addresses this way we mimic a Moore neighbourhood in a 2D grid.

```

neighbors:Address[ 8] <-> [ 8] Address:
  neighbors;
address:Address[ 1] <-> [ 0-] Person:
  inhabitants;

```

Modelling the social environment

Besides the geographical, physical space, individuals are also positioned in 'social space'. This is modelled by connecting each *Person* agent to a *Household*, that is, a family whose members share their resources.

Households are modelled as agents and are attributed with their capital.

```
Household(
  capital: real := 0
);
household:Household[ 1] <->[ 1-] Person:
  members;
```

But individuals are not only related to other individuals by sharing the same household. Person agents are also linked in a network structure of family, partnership, and friendship ties. Friends are modelled by many-to-many links between individuals, while being married is modelled by relating two individuals to each other. Each Person also has two parents, whereas not every individual has children. The parents–children link indirectly defines siblings: the siblings of a Person agent are the children of their parents apart from themselves. This exemplifies why dead agents can still be important for the model, although they do not act by themselves any more.

```
friends:Person[ 0-] <->[ 0-] Person:
  friends;
parents:Person[ 2] <->[ 0-] Person:
  children;
partner:Person[ 0-1] <->
  [ 0-1] Person:partner;
```

Modelling the migration decision process

Within the migration process, the following stages are distinguished: intention formation, planning, preparation, and migration. At each stage, different options are viable. In the preparation stage, for example, the agent can proceed to actually migrate. Alternatively, their migration intention may have been weakened (e.g., by stricter border control) so much that it becomes negative and the decision process is abandoned. Proceeding through the decision stages competes with further options that are not part of the decision process, but simply might happen, such as death or childbirth. Everything that can happen to an agent is defined in terms of concurrent behaviour rules, whose conditions determine which rules compete, and whose rates determine the likelihood of seeing the described behaviour within a certain time interval. The different rate expressions are responsible for obtaining the waiting time for the different rules stochastically or deterministically. Only the rule for

which the shortest waiting time has been computed is executed, and all rules are re-evaluated in the new state. Please note that all rates, factors, and calculations are taken from the original model realized in NetLogo and described in Klabunde et al. 2017 (in this Supplement).

```
Person
| ego.migrationIntention() >= 0,
  ego.migrationStage = "preparation"
@ ego.migrationAdvancementRate() *
  (1 - ego.borderEnforcement
    Factor())
-> ego.migrationStage := "no
  intention,"
  ego.migrationAttempts += 1,
  ego.failedMigrationAttempts += 1;

| ego.migrationIntention() >= 0,
  ego.migrationStage = "prepara
    tion," ego.canAffordMigration()
@ ego.migrationAdvancementRate() *
  ego.borderEnforcementFactor()
-> ego.migrate();

@ ego.mortalityModifier *
  ego.mortalityRate()
-> ego.dieAndRemove();
```

This listing shows three rules that are defined for Person agents. The first two rules apply to agents in the preparation stage who have a positive migration intention, and thus try to migrate. The first rule describes an unsuccessful migration attempt, while the second one describes a successful migration. The third rule, modelling the death of agents, has no conditions and applies to all Person agents. Functions are used to define exponentially distributed waiting times for all three rules.

Thus, for each Person whose attribute migrationStage is 'preparation', these behaviour rules (among others) are active and competing. As detailed above, waiting times for these rules—advancing and resetting the migration process and dying—are obtained by evaluating their waiting time expressions. Only the rule with the shortest waiting time is actually executed. This stochastic race naturally models the competition between different behaviours of an agent. The nondeterminism in the decision process is solved through competition between the rules. Finally, not only do the rules for a single agent compete, but the rules for all active agents of any type in the model also compete.

Functions—determining values and constraining stochastic race

As shown in the previous subsection, functions facilitate the compact definition of stochastic waiting times. The resulting stochastic race of competing risks resolves the conflict between different behavioural options of an agent. Thus, functions play a central role in modelling decision processes according to the TPB. For example, among the core elements of the migration model are the calculations of an agent's migration intention and the resulting migration advancement rate. They can be expressed as ML3 functions:

```
Person.migrationIntention() :=
  ?alpha * ?MA + ?beta * ?SN + ?gamma * ?PBC
  where ?alpha := attitudeWeight,
         ?beta := socialNormsWeight,
         ?gamma := perceivedBehavioral
           ControlWeight,
         ?MA := ego.migrationAttitude(),
         ?SN := ego.socialNorms(),
         ?PBC := ego.perceivedBeha
           vioralControl();

Person.migrationAdvancementRate() :=
  ?rho * e^(?theta * ego.migration
    Intention())
  where ?rho := advancementRate
    Baseline,
         ?theta := advancement
           RateIntentionWeight;
```

Whereas the weighting factors are fixed model parameters, migration attitude, social norms, and perceived behavioural control are themselves calculated by functions. This delegation of calculations resembles the way the model is detailed when using natural language (see equation (2)). The migration advancement rate is also calculated exactly as in the model description (see equation (3)). It uses the function for the migration intention and is itself used in the waiting time expressions of the decision process rules stated earlier. Thus, the executable model description in ML3 closely resembles the mental model the demographer wants to simulate.

Experimentation

Using this implementation of the migration model in ML3 and a prototypical implementation of the completely model-independent simulation algorithm, we

conducted similar experiments to Klabunde et al. 2017 (in this Supplement). One of the properties of interest was the age profile of migrants at the time of migration (Figure 3). Currently, the simulator only allows certain kinds of macro-level observation during simulation, for example, population size at different times or the distribution of an attribute in a population of agents. In future, we plan to provide more powerful means for observation, as well as experimentation support, for example, by implementing a binding to SESSL, a domain-specific language that supports the modeller in the execution of complex simulation experiments (Ewald and Uhrmacher 2014). This would enable systematic experimentation with models, using methods such as parameter optimizations or sensitivity analysis. Rudimentary SESSL support for ML3 is already available online (<http://sessl.org>) and in the supplementary material. Further exploration of the experimentation support is the subject of future work.

Discussion of the features of ML3

Having showcased some features of ML3 and their usefulness for a specific application, we now discuss some more general aspects of the language design.

Attributes, conditions and constraints for concurrent processes

The modelling paradigm that ML3 uses—agents acting as parallel processes—strongly resembles process algebra approaches. One of the most commonly known examples of this family is the π -calculus (Milner et al. 1992), which sparked some work on enriching processes with stochastic rates (Priami 1995) and, later on, attributes (John et al. 2008; Bortolussi et al. 2015). Whereas ML3 shares its concept of parallel, attributed processes interacting stochastically in continuous time with these approaches, it uses a different concept of interaction. In process algebra, processes typically communicate and thus interact synchronously or asynchronously over dynamically created channels. ML3, however, structures its acting agents into a network of links, which may continue to exist unchanged, even after the death of an agent. However, links can still be created, modified, and deleted dynamically as a result of the actions of an agent. This emphasizes the anchoring of individuals in a changing environment or context, and facilitates statements about the interdependence of the behaviour of agents and their social networks.

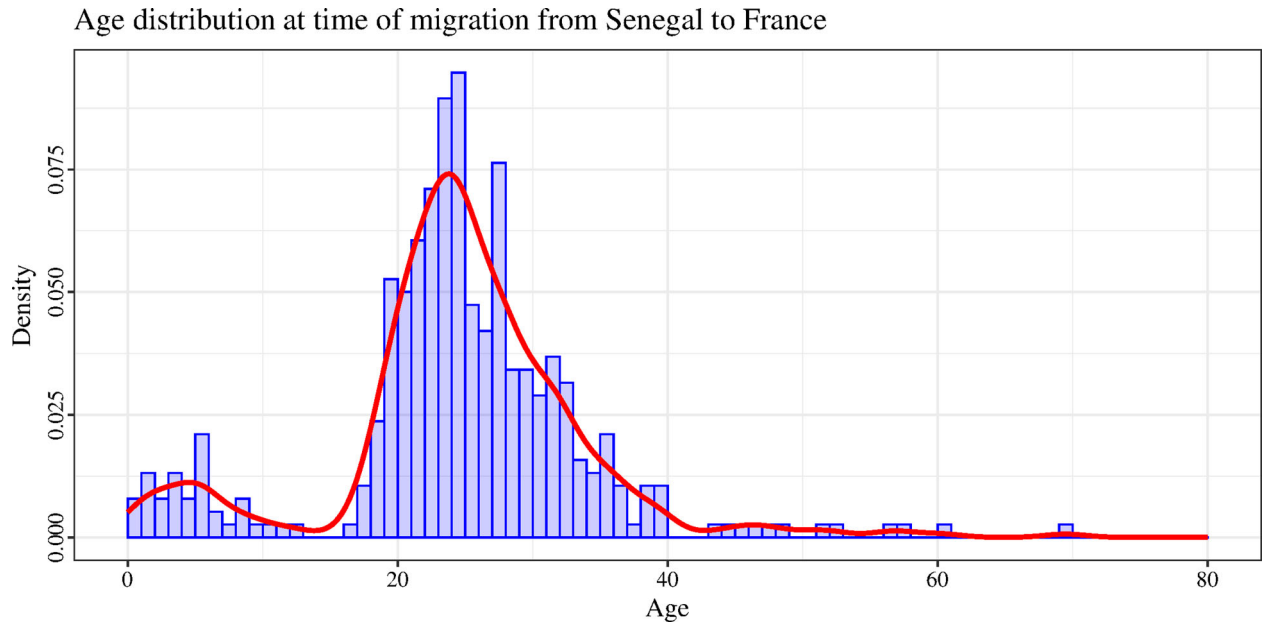


Figure 3 Distribution of ages at migration to the host country from a single simulation run. In this configuration, each person starts to consider migration at a different age drawn from a normal distribution (see Klabunde et al. 2017 for details)

Context of demographic agents—networks of interactions rather than a 2D grid

The surroundings of an agent in ML3 are entirely represented by their links to other agents. Instead of using a grid to model the position of an agent in space, their proximity to other agents is modelled by connections in a network. As multi-agent simulation packages like NetLogo demonstrate, grid spaces support an easy animation of the entities' behaviour in 2D. However, complex models with agents related in several dimensions of geographical and, particularly, social space benefit from the more flexible network approach. Therefore, multi-agent simulation packages such as Mason (Luke et al. 2005) support network spaces in addition to grid spaces. To mimic the 2D grid of the original NetLogo model, we describe a Moore neighbourhood by linking each address with eight neighbours. Such a graph-based neighbourhood model is easily extendable to other concepts of geographical space.

Imperative modelling

For ML3, we selected an imperative style of model programming. This is different from languages we have developed previously. In ML-Rules, for example, reactions with reactants, products, and a reaction rate are the objects of interest. The

description of these typically requires no complicated calculations. In ML3, however, the entities or processes are the focus of interest. The updating of their values, or transitions between the different stages of life, is determined by complex decision rules. This focus on entities and their individual states and state changes is reflected in the imperative style of programming, which the demographic modeller is also typically familiar with.

External vs. internal domain-specific languages

Domain-specific languages are subdivided into: (1) internal or embedded domain-specific languages, which build on a general programming language as their host language; and (2) external domain-specific languages, which are shipped with their own compiler. Both approaches have their pros and cons. Embedded languages are typically more flexible, as they allow the user to fall back on the functionality of the host language. However, they do not allow as much freedom of choice when it comes to the concrete syntax design as external domain-specific languages do (van Deursen et al. 2000). Our current realization has been produced as an external language, as we wanted to be able to discuss and evaluate different alternatives of the concrete syntax freely. However, one drawback of an external

domain-specific language is that the models defined in such a language are not accessible for optimization by the compiler of the general-purpose programming language the simulation is executed in. Exploiting such optimization and tools of a host language is one of the pros of realizing a modelling language as an embedded or internal domain-specific language (Zunino et al. 2015). Efficient simulation is somewhat difficult when modelling concurrent complex decisions by stochastic race. Thus, moving from an external to an internal domain-specific model language design, for example, based on Scala, is an option that will be explored in the future. We have recently proposed an extension for Repast Symphony (a framework for agent-based modelling and simulation), which allows the definition of simple continuous-time agent-based models in a manner that is similar to ML3 (Warnke et al. 2016).

Complex decision processes and stochastic race

Complex decisions are modelled by functions based on the situation and context of each agent. As already found in the design of other modelling languages, the ability to define arbitrary functions on the attributes, structure, and components of models is an essential feature of expressive modelling languages (John et al. 2011; Warnke, Helms et al. 2015). Given different options, the functions are used to calculate the respective attractiveness to an agent of deciding on a particular option and to translate this attractiveness into corresponding transition times. This leads to a compact description of the decision process as a choice among different competing options. At the same time, however, it requires a lot of calculation effort for the simulation algorithm. The simulator calculates the transition time for all options, and, as is common with competing risks, the option with the earliest transition time is selected, which then precludes the occurrence of the other transitions. This is done for all agents and again the option with the smallest waiting time is selected. Our current implementation recalculates the transition time after a transition has been executed, for all options and all agents. An alternative would be to maintain a dependency graph of options (and the corresponding transitions) that would allow us to distinguish between transition times that need to be recalculated and those that are still valid after a transition has taken place. Exploring this alternative will be the subject of future work: not only to speed up simulation, but also to support a wider range of

probability distributions for modelling sojourn times. A first improved, more efficient simulation algorithm for ML3 has already been developed (Reinhardt and Uhrmacher 2017).

Plausible initial states for complex models

The generation of an initial state for a micro-model is much more complex than for a macro-model. For example, if some data about the population structure are available, a number of aggregate variables for a macro-model can be derived directly. In an agent-based model, however, a consistent population of attributed agents with the same properties as the data needs to be created. In ML3, the links between the agents also need to be set plausibly. For example, a model could require that children are at least twelve years old, but are no more than 60 years younger than their parents. Such constraints must be considered when creating the initial population for a simulation run.

One way to create a population with plausible structures is to prepend a warm-up phase to the simulation run (Noble et al. 2012). This way, the initial population for the actual experiment is generated by the simulation itself and is thus consistent. However, it is also stochastically created and, as such, not entirely controllable. This makes it impossible to create initial states with specific desired properties, for example, derived from data. We are currently investigating methods to construct plausible and consistent ML3 agent populations and links from survey data.





Conclusions

In this paper, we have demonstrated the application of the domain-specific modelling language for agent-based computational demography, ML3, to models of decision processes. Although these processes can be highly complex, for example, being conditional on agent attributes or time-varying input data, they can be described succinctly with ML3. Based on an example model, we have discussed the features of the language design of ML3, such as the concurrent process metaphor, the integration of agents into a link network, and the modelling of nondeterminism via stochastic race. Future work will include improvements to the simulation algorithm, as well as to the language itself, to pave the way for conducting simulation studies efficiently.

Notes and acknowledgements

- 1 Please direct all correspondence to Tom Warnke, Modeling and Simulation Group, Institute of Computer Science, Albert-Einstein-Str. 22, 18059 Rostock, Germany; or by E-mail: tom.warnke@uni-rostock.de
- 2 This research was supported by the German Research Foundation (DFG) under Grant UH-66/15-1 (MoSiLLDe).

ORCID

Tom Warnke  <http://orcid.org/0000-0002-8196-2943>
 Anna Klabunde  <http://orcid.org/0000-0001-6217-8072>
 Frans Willekens  <http://orcid.org/0000-0001-6125-0212>
 Adelinde Uhrmacher  <http://orcid.org/0000-0001-5256-4682>

Supplemental data

The complete migration model file and a small experimentation tool are available in the supplementary material, which can be found at: <https://doi.org/10.1080/00324728.2017.1380960>

References

- Ajzen, Icek. 1991. The theory of planned behavior, *Organizational Behavior and Human Decision Processes* 50(2): 179–211.
- Ajzen, Icek and Jane Klobas. 2013. Fertility intentions: an approach based on the theory of planned behavior, *Demographic Research* 29: 203–232.
- Billari, Francesco C., Alexia Prskawetz, Belinda A. Diaz, and Thomas Fent. 2007. The ‘wedding-ring’: an agent-based marriage model based on social interaction, *Demographic Research* 17: 59–82.
- Bortolussi, Luca, Rocco de Nicola, Vashti Galpin, Stephen Gilmore, Jane Hillston, Diego Latella, Michele Loreti, and Mieke Massink. 2015. CARMA: collective adaptive resource-sharing Markovian agents. *Electronic Proceedings in Theoretical Computer Science* 194: 16–31.
- Dijkstra, Edsger W. 1975. Guarded commands, nondeterminacy and formal derivation of programs, *Communications of the ACM* 18(8): 453–457.
- Doob, Joseph L. 1945. Markoff chains—denumerable case, *Transactions of the American Mathematical Society* 58(3): 455–473.
- Ewald, Roland and Adelinde M. Uhrmacher. 2014. SESSL: a domain-specific language for simulation experiments, *ACM Transactions on Modeling and Computer Simulation* 24(2): 1–25.
- Feng, Cheng and Jane Hillston. 2014. PALOMA: A process algebra for located Markovian agents. *Quantitative evaluation of systems*, Florence, Italy: Springer, 2014, pp. 265–280.
- Fent, Thomas, Belinda A. Diaz, and Alexia Prskawetz. 2013. Family policies in the context of low fertility and social structure, *Demographic Research* 29: 963–998.
- Gillespie, Daniel T. 1976. A general method for numerically simulating the stochastic time evolution of coupled chemical reactions, *Journal of Computational Physics* 22(4): 403–434.
- Godin, Gaston and Gerjo Kok. 1996. The theory of planned behavior: a review of its applications to health-related behaviors, *American Journal of Health Promotion* 11(2): 87–98.
- Grimm, Volker, Uta Berger, Donald L. DeAngelis, J. G. Polhill, Jarl Giske, and Steven F. Railsback. 2010. The ODD protocol: a review and first update, *Ecological Modelling* 221(23): 2760–2768.
- Grönninger, Hans, Holger Krahn, Bernhard Rumpe, Martin Schindler, and Steven Völkel. 2007. *Textbased Modeling*. Mainz: Johannes-Gutenberg-Universität.
- Henzinger, Thomas A., Barbara Jobstmann, and Verena Wolf. 2011. Formalisms for specifying markovian population models, *International Journal of Foundations of Computer Science* 22(04): 823–841.
- Hills, Thomas and Peter Todd. 2008. Population heterogeneity and individual differences in an assortative agent-based marriage and divorce model (MADAM) using search with relaxing expectations, *Journal of Artificial Societies and Social Simulation* 11(4): 5.
- Jansen, A. P. J. 1995. Monte Carlo simulations of chemical reactions on a surface with time-dependent reaction-rate constants, *Computer Physics Communications* 86(1): 1–12.
- John, Mathias, Cédric Lhoussaine, Joachim Niehren, and Adelinde M. Uhrmacher. 2008. The attributed Pi calculus. *Computational methods in systems biology*, Berlin, Heidelberg: Springer, 2008, pp. 83–102.
- John, Mathias, Cédric Lhoussaine, Joachim Niehren, and Adelinde Uhrmacher. 2010. The attributed Pi calculus with priorities, *Transactions on Computational Systems Biology* 5945(12): 13–76.
- John, Mathias, Cédric Lhoussaine, Joachim Niehren, and Cristian Versari. 2011. Biochemical reaction rules with constraints. *20th European Symposium on Programming*, Saarbrücken, Germany: Springer, 2011, pp. 338–357.
- Klabunde, Anna and Frans Willekens. 2016. Decision-making in agent-based models of migration: state of the art and challenges, *European Journal of Population* 32(1): 73–97.

- Klabunde, Anna, Sabine Zinn, Matthias Leuchter, and Frans Willekens. 2015. *An Agent-Based Decision Model of Migration, Embedded in the Life Course—Model Description in ODD+ D Format*. Rostock, Germany: Max Planck Institute for Demographic Research.
- Klabunde, Anna, Sabine Zinn, Frans Willekens, and Matthias Leuchter. 2017. Multistate modelling extended by behavioural rules: an application to migration, *Population Studies* 71(S1): S51–S67. doi:10.1080/00324728.2017.1350281.
- Kravari, Kalliopi and Nick Bassiliades. 2015. A survey of agent platforms, *Journal of Artificial Societies and Social Simulation* 18(1): 11.
- Luke, Sean, Claudio Cioffi-Revilla, Liviu Panait, Keith Sullivan, and Gabriel Balan. 2005. MASON: A multi-agent simulation environment, *Simulation* 81(7): 517–527.
- Maus, Carsten, Stefan Rybacki, and Adelinde M. Uhrmacher. 2011. Rule-based multi-level modelling of cell biological systems, *BMC Systems Biology* 5(1): 166.
- Milner, Robin, Joachim Parrow, and David Walker. 1992. A calculus of mobile processes, I, *Information and Computation* 100(1): 1–40.
- Noble, Jason, Eric Silverman, Jakub Bijak, Stuart Rossiter, Maria Evandrou, Seth Bullock, Athina Vlachantoni, and Jane Falkingham. 2012. Linked lives: the utility of an agent-based approach to modeling partnership and household formation in the context of social care. *Winter simulation conference 2012*, Berlin, Germany: Winter Simulation Conference, pp. 93:1–93:12.
- Pârvu, Ovidiu, David Gilbert, Monika Heiner, Fei Liu, Nigel Saunders, and Simon Shaw. 2015. Spatial-temporal modelling and analysis of bacterial colonies with phase variable genes, *ACM Transactions on Modeling and Computer Simulation* 25(2): 13:1–13:25.
- Petre, Marian. 1995. Why looking isn't always seeing: readership skills and graphical programming, *Communications of the ACM* 38(6): 33–44.
- Philipov, Dimiter, Aart C. Liefbroer, and Jane E. Klobas. 2015. *Reproductive Decision-Making in a Macro-Micro Perspective*. Dordrecht: Springer Netherlands.
- Priami, C. 1995. Stochastic π -calculus, *The Computer Journal* 38(7): 578–589.
- R Core Team. 2015. *R: A language and environment for statistical computing*. Available: <https://www.R-project.org/>. Vienna, Austria (accessed: September 2017)
- Reinhardt, Oliver and Adelinde M. Uhrmacher. 2017. An efficient simulation algorithm for continuous-time demographic agent-based models. *Spring simulation multi-conference*, Virginia Beach, VA, USA 2017.
- Sheppard, Colin J., and Steve Railsback. 2015. *Time extension for NetLogo (Version 1.2) [Software]*.
- Steiniger, Alexander, Adelinde M. Uhrmacher, Sabine Zinn, Jutta Gampe, and Frans Willekens. 2014. The role of languages for modeling and simulating continuous-time multi-level models in demography. *Winter simulation conference*, Savannah, GA, USA: IEEE Press, 2014, pp. 2978–2989.
- Theodoropoulos, Georgios, Rob Minson, Roland Ewald, and Michael Lees. 2009. Simulation engines for multi-agent systems. *Multi-agent Systems. Simulation and Applications*. Boca Raton, Fla.: Taylor & Francis, pp. 77–108.
- Todd, Peter M., Francesco C. Billari, and Jorge Simao. 2005. Aggregate age-at-marriage patterns from individual mate-search heuristics, *Demography* 42(3): 559–574.
- Train, Kenneth. 2009. *Discrete Choice Methods with Simulation*. Cambridge: Cambridge University Press.
- Van Deursen, Arie, Paul Klint, and Joost Visser. 2000. Domain-specific languages: an annotated bibliography, *ACM SIGPLAN Notices* 35(6): 26–36.
- Walsh, Stephen J., George P. Malanson, Barbara Entwisle, Ronald R. Rindfuss, Peter J. Mucha, Benjamin W. Heumann, Philip M. McDaniel, Brian G. Frizzelle, Ashton M. Verdery, Nathalie E. Williams, Xiaozheng Yao, and Deng Ding. 2013. Design of an agent-based model to examine population-environment interactions in Nang Rong district, Thailand, *Applied Geography* 39: 183–198.
- Warnke, Tom, Anna Klabunde, Alexander M. Steiniger, Frans Willekens, and Adelinde Uhrmacher. 2015. ML3: a language for compact modeling of linked lives in computational demography. *Winter simulation conference*, Huntington Beach, CA, California, 2015.
- Warnke, Tom, Tobias Helms, and Adelinde M. Uhrmacher. 2015. Syntax and semantics of a multi-level modeling language. *3rd ACM SIGSIM conference on principles of advanced discrete simulation*, London, UK. New York, NY, USA: ACM, 2015. pp. 133–144.
- Warnke, Tom, Oliver Reinhardt, and Adelinde M. Uhrmacher. 2016. Population-based CTMCS and agent-based models. *Winter simulation conference*, Arlington, VA, USA, 2016, pp. 1253–1264.
- Wilensky, Uri. 1999. *NetLogo*. Evanston, IL: Center for Connected Learning and Computer-Based Modeling, Northwestern University.
- Zunino, Roberto, Đurica Nikolić, Corrado Priami, Ozan Kahramanoğulları, and Tommaso Schiavinotto. 2015. L — An imperative DSL to stochastically simulate biological systems. *Programming Languages with Applications to Biology and Security. Essays Dedicated to Pierpaolo Degano on the Occasion of His 65th Birthday*. Cham: Springer International Publishing, pp. 354–374.