Variable	Description	Value
NumComm	Number of communities	500
NumUser	Number of users	500
NumTask	Number of tasks	500
MinP	Minimum task number	4
MaxP	Maximum task number	100
i	Community index	i = 1, 2,, NumComm
j	User index	j = 1, 2,, NumUser
k	Task index	k=1,2,,NumTask
s	User skill	$2 \le s \le \sqrt{MaxP}$
P_i	User population of community i	$0 \le P_i \le NumUser$
T_i	Topic of community i	$MinP \le T_i \le MaxP$
A_j	Attention level of user j	$A_j = 0, 1$
S(j)	Skill of user j	$2 \le S(j) \le \sqrt{MaxP}$
UC_j	Community that user j belongs to	$UC_j \in i$
TC_k	Community that task k belongs to	$UT_k \in i$
TN_k	Task number of task k	$MinP \le TN_k \le MaxP$
$CH_{k,s}$	Record of skill s being applied to task k	$CH_{k,s} = 0, 1$
rr	Community replacement rate	0.05
λ	Topic adjustment rate	0.1

Table 1: This table summarizes the variables and parameters of the model. Parameter values are chosen ad hoc and are provided only for completeness.

1 Introduction

In order to study the population dynamics of online communities, a simulation model is developed. In this model, users can move between communities while completing tasks that they encounter. The model includes: task generation, task allocation, user contribution, task completion, and user movement.

Three types of agents are considered in this model: tasks, users, and communities. Tasks are described by a task number, a list of past contributions, and a community to which that task belongs. Users are described by a skill number, their attention level, and the community they are participating in. Communities have a topic as well as user and task populations. The model is run in discrete time and the state variables are modified at each time step. Table 1 provides a complete list of variables and parameters.

The model used is a foraging model, tailored to suit online communities. Users are the foragers and they forage for tasks. However, users are heterogeneous, and a task that one user can complete might be outside the skill set of another user. Thus, tasks can be thought of as the resource users are foraging for, and users contributing to a task is what consumes that resource. Communities serve to divide the simulation environment into locations for foragers to move between.

2 Tasks

The task considered in this model is the identification of prime numbers. This task was chosen because it allows for heterogeneity in task "topic, as well as heterogeneity in agent skill. A user's skill is thought of as an integer. This integer is the number the user can divide by. A task is complete when a user successfully divides the task (nonprime) or when every applicable skill has been applied to the task (prime). Tasks are the resource that users forage for. During each time step of the model tasks must be generated, allocated to communities, receive user contributions, and checked for completion.

In this model task generation occurs separately from task allocation. It is assumed that tasks are generated independently of the communities modeled. This could be justified by the observation that in online communities, those that ask questions, and those that answer them, are largely disjointed. Newly generated tasks then occur according to the parameters of the model rather than the dynamic variables. When a new task is generated it is chosen randomly to be any of the possible tasks considered (integers between MinP and MaxP).

$$P\left(TN_{\{k|TC_k(t)=0\}}(t+1)=x\right) = \left\{ \begin{array}{c} \frac{1}{1+MaxP-MinP} \\ 0 \end{array} \middle| \begin{array}{c} x \in [MinP, MaxP] \\ x \notin [MinP, MaxP] \end{array} \right\}$$

Created tasks must be assigned to a community. Tasks are assigned based on two key factors: the size of the community (P_i) and the topic of the community (T_j) . Community size is included to mimic the effect that popular communities receive more tasks then less popular communities. This is a form of preferential attachment, which is often included in models of online communities. Communities are assumed to focus on specific topics, and tasks are allocated to communities with similar topics. The topic of a community is not static, but reflects the tasks that a community has recently had success with.

$$P\left(TC_{\{k|TC_{k}(t)=0\}}(t+1)=x\right) = \left\{ \begin{array}{c} \frac{1}{\sum_{i}f(i,k)} & f(x,k)=1\\ f(x,k)\neq 1 \end{array} \right\}$$
$$f(i,k) = \left\{ \begin{array}{c} 1 & TN_{k}(t+1) \in \left[T_{i}(t) - \frac{100P_{i}}{NumUser}, T_{i}(t) + \frac{100P_{i}}{NumUser}\right]\\ TN_{k}(t+1) \notin \left[T_{i}(t) - \frac{100P_{i}}{NumUser}, T_{i}(t) + \frac{100P_{i}}{NumUser}\right] \end{array} \right\}$$

Although the tasks that enter the simulation environment are independent of the communities, which community receives the task does depend on both the size and previous work of the community. Communities are considered to have a topic, bounded in the same domain as tasks, that reflects the specialty of the community. Additionally, larger communities are considered more able to meet a variety of challenges, expanding the range of tasks they can accept. The growth in range is assumed to be linear and such that if all users are in a single community, that community can accept all tasks. The new task generation and allocation occurs once per task completed in the previous round. This way, a population of NumTask tasks is maintained in the simulation environment.

Once a task is assigned to a community, its users are able to contribute to it. In a given community, the users that contribute, and the tasks that get contributions, depend on the number of users with a particular skill and the number of tasks that require that skill. For each community and skill the number of users with that skill, and the number of tasks to which that skill is applicable, are counted. This can be represented as in:

$$NU(i,s) = \sum_{j|UC_{j}(t)=i,S(j)=s} 1$$
$$NT(i,s) = \sum_{k|UT_{k}(t)=i,TN_{k}(t+1) \le s^{2},CH_{k,s}(t)=0} 1.$$

Only unique contributions are counted, and users that are able to make a unique contribution are assumed to do so. These assumptions mean that for a given community and skill, either every user will contribute or every task will receive a contribution. Of the larger population, a number of members equal to that of the smaller population is chosen to provide or receive the contributions. These assumptions provide probabilities for a contribution occurring for both tasks and users where the probability of a task receiving a contribution is determined by the relative abundance of users and vice versa.

$$P\left(CH_{\{k,s|CH_{k,s}(t)=0,TN+k\leq s^2\}}(t+1)=1\right) = \begin{cases} \min\left(1,\frac{NU(TC_k(t+1),s)}{NT(TC_k(t+1),s)}\right) & NT(TC_k(t+1),s) > 0\\ 0 & NT(TC_k(t+1),s) \leq 0 \end{cases}$$

$$P\left(A_j(t+1)=1\right) = \min\left(1,\frac{NT(TC_k(t+1),s)}{NU(TC_k(t+1),s)}\right)$$

A trivial consequence of these equations is that for each time-step, in each community, either every user with a particular skill will contribute or every task that requires that skill will have it applied.

Completed tasks are removed from the simulation at the end of every timestep, rather than as they are completed. This can be thought of as time used to verify that the task is complete or to accept a solution. The list of skills applied to a task is used to check if either a user has been able to identify the task as nonprime, or the community has identified a number as prime:

$$Nonprime(k) = \left\{ \begin{array}{c} 1\\ 0 \end{array} \middle| \begin{array}{l} \{s|TN_k(t+1), CH_{k,s}(t+1) = 1\} \neq \emptyset\\ \{s|TN_k(t+1), CH_{k,s}(t+1) = 1\} = \emptyset \end{array} \right\}$$
$$Prime(k) = \left\{ \begin{array}{c} 1\\ 0 \end{array} \middle| \begin{array}{l} 1 + \sum_s CH_{k,s} > \sqrt{TN_k}(t+1)\\ 1 + \sum_s CH_{k,s} \le \sqrt{TN_k}(t+1) \end{array} \right\}$$

Complete(k) = max(NonPrime(k), Prime(k))

Nonprime numbers are detected by checking if any of the contributions to a task divides the task number, if at least one does the number cannot be prime. The identification of prime numbers requires that every skill that is applicable (less than or equal to the square root of the task) has been applied. It is possible for the function identifying primes to give false positives but, as no distinction is made, it does not affect results. Tasks that are completed need to be removed for the model to allow for new tasks. This is accomplished by the following equations:

$$CH_{\{k,s|Complete(k)=1\}}(t+1) = 0$$

 $TC_{\{k|Complete(k)=1\}}(t+1) = 0$

The other effect of a completed task is the shift of topic in the communities, which is accomplished with a simple learning algorithm of the form:

$$T_i = (1 - \lambda) T_i + \lambda T N_k$$

where λ is a topic adjustment rate, T_i is the topic of community *i*, and N_k is the task number of task *k*.

3 Users

The users of online communities are a heterogeneous population searching for something to hold their attention. Users differ in the skill they have for completing tasks, as well as in location. The skill of users is a fixed initial condition so users must move between communities to find tasks to contribute to. Users can leave a community either to join another community or to establish a new community.

Movement of users is driven by an algorithm based on win-stay, lose-shift, as well as preferential attachment. Users are assumed to change communities every round in which they fail to contribute, and where they move to is based on the population of the other communities:

$$P\left(UC_{\{j|A_j=0\}}(t+1)=x\right) = \left\{ \begin{array}{c} \frac{P_x}{\sum_{i\neq x} P_i} & x\neq UC_j(t)\\ 0 & x=UC_j(t) \end{array} \right\}$$

The other condition for user movement is the foundation of a new community, and it is assumed to increase linearly with the number of abandoned communities:

$$P\left(UC_{j}(t+1)=x\right) = \left\{ \begin{array}{c} \frac{rr}{NumUser} \\ 0 \end{array} \middle| \begin{array}{c} P_{x}=0 \\ P_{x}\neq 0 \end{array} \right\}$$

4 Initialization

At initialization users are assigned a skill between two and the square root of MaxP, which is the set of possible divisors of task numbers. Communities are assigned a topic from the same distribution as task numbers, and users are assigned to communities. Each of these assignments are assumed to be random.

5 Implementation

The model was implemented in both MATLAB and NetLogo.