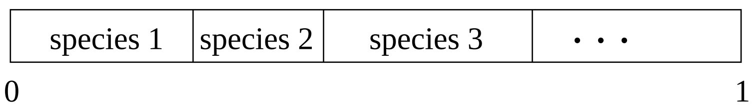
14 Evolutionary Systems II

In this section we want to study how powerful linear and quadratic dynamical systems are in solving problems. For this we will assume that they act on an infinitely large population. Viewing the state space S as the set of all possible species, we can repesent every population as a vector $\boldsymbol{q}$ where $q_i \in [0,1]$ represents the relative share of species i (i.e. individuals of type i), and $\sum_{i \in S} q_i = 1$.



To ensure that the outcome of a dynamical system on an initial distribution of shares always represents a valid distribution of shares (i.e., $\sum_{i \in S} q_i = 1$), we will only allow the use of stochastic transition matrices. Furthermore, to ensure that each component of a dynamical system is efficiently computable, we demand for any given dynamical system that it is *succinctly-defined.* A linear/quadratical system is succinctly-defined if both its initial distribution and transition matrix are succinctly-defined. Furthermore, a distribution $\boldsymbol{q}$ is *succinctly-defined* if there is a probabilistic polynomial time Turing machine which produces a sample i according to the distribution $\Pr[i] = q_i$. Similarly, a stochastic transition matrix $\boldsymbol{P} = (p_{i,j})$ is called *succinctly-defined* if there is a probabilistic polynomial time Turing machine which, on input i, can produce a sample j according to the distribution $\Pr[j] = p_{i,j}$. Given a succinctly-defined linear/quadratic system, we want to investigate the following sampling problem:

Sampling problem:

- **Input:** $|S|$ (in binary notation), a succinctly-defined dynamical system D given as two Turing machines M_q and M_P (one for the initial distribution, and one for the transisiton matrix), 1^t (a chain of t ones), and an error parameter ϵ.
- **Output**: A sample $x \in S$ chosen according to a distribution $\boldsymbol{q}'$ with $||\boldsymbol{q}' - \boldsymbol{q}^{(t)}||_1 \leq \epsilon$, where $\boldsymbol{q}^{(t)}$ is the distribution of shares of D after t rounds.

We note here that since we assume the population to be of infinite size, the share of species i after t rounds is equal to $\boldsymbol{q}^{(t)}$ even though we originally have probabilistic transitions. This is due to the law of large numbers. Hence, we can view $\boldsymbol{q}^{(t)}$ as truely representing the share of species i after t rounds.

Suppose we had a probabilistic poynomial time Turing machine that solves this sampling problem. Then this Turing machine could be used to approximate the distribution of shares after t rounds of the dynamical system with high accuracy, and therefore the dynamical system would not be more powerful than a probabilistic Turing machine.

In order to speficy what we mean by "powerful", we will introduce notation that allows us to specify which classes of problems can be solved efficiently by linear/quadratic dynamical systems. Let φ be a mapping from $\{0,1\}^*$ to the set of all vectors that represent probability distributions.

We say that φ is *succinctly-defined* if there is a probabilistic polynomial time Turing machine which, on input $x \in \{0,1\}^*$, can produce a sample i according to the distribution $\Pr[i] = \varphi(x)_i$. Furthermore, a pair $(\varphi, \boldsymbol{P})$ is called *succinctly-defined* if both the mapping φ and the transition matrix $\boldsymbol{P}$ are succinctly-defined. Each pair $(\varphi, \boldsymbol{P})$ establishes a collection of dynamical systems D_x that depend on input x. If $(\varphi, \boldsymbol{P})$ represents a collection of linear (reps. quadratic) dynamical systems, we call it a *uniform family of linear (resp. quadratic) dynamical systems*. Let $\mathcal{LDS}$ be the class of all succinctly-defined uniform families of linear dynamical systems, and $\mathcal{QDS}$ be the class of all succinctly-defined uniform families of quadratic dynamical systems.

A decision problem L is said to be in $\mathrm{P}(\mathcal{LDS})$ (resp. $\mathrm{P}(\mathcal{QDS})$) if and only if there is a family of linear (resp. quadratical) dynamical systems $\mathcal{D} = (\varphi, \boldsymbol{P})$ in this class and a polynomial p so that for any input x, $\mathcal{D}$ acting on the initial distribution $\varphi(x)$ of species needs at most $p(|x|)$ rounds before it holds:

- if $x \in L$, then the share of individuals that confirm $x \in L$ is at least $2/3$, and
- if $x \notin L$, then the share of individuals that confirm $x \in L$ is at most $1/3$.

Obviously, if for every succinctly-defined LDS (resp. QDS) the sampling problem above can be solved in polynomial time, then this would mean that $\mathrm{P}(\mathcal{LDS})$ (resp. $\mathrm{P}(\mathcal{QDS})$) is a subset of BPP. Thus, we will concentrate in the following on studying the complexity of the sampling problem for succinctly-defined linear and quadratic dynamical systems.

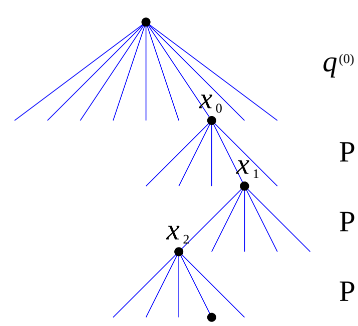
14.1 Linear dynamical systems

We will use here instead of a stochastic dynamical system again the notion of a Markov chain. The following result shows that Markov chains are easy to simulate via Turing machines.

Theorem 14.1 *For any succinctly-defined Markov chain, the sampling problem can be solved by a probabilistic Turing machine in polynomial time.*

Proof. For any $x \in S$, let e_x denote the vector that has a 1 at the xth position and a 0 everywhere else. We use the following Turing machine:

choose a sample x_0 according to the initial probability distribution $\boldsymbol{q}^{(0)}$
for $i = 0$ to $t-1$:
  choose a sample x_{i+1} according to the probab. distribution given by $\boldsymbol{p}_{x_i} = (p_{x_i,1}, \ldots, p_{x_i,s})$
output x_t



Obviously, the Turing machine requires only polynomially many time steps. It remains to prove that this Turing machine generates x_t with the desired probability distribution. This will be done by induction on the number of rounds of the algorithm.

From the construction of the algorithm we know that x_0 is chosen according to the distribution $(\boldsymbol{q}^{(0)})^T \cdot \boldsymbol{P}^0$. Hence, for $t = 0$ the theorem is true. So suppose now that x_i is chosen according to the distribution $(\boldsymbol{q}^{(0)})^T \cdot \boldsymbol{P}^i$. Then we want to show that x_{i+1} is chosen according to the distribution $(\boldsymbol{q}^{(0)})^T \cdot \boldsymbol{P}^{i+1}$. This would conclude the induction and prove the theorem for all t. According to the algorithm, given an x_i, the probability that $x_{i+1} = x$ for some species x is equal to $\boldsymbol{p}_{x_i}$. Hence,

$$\begin{aligned}
\Pr[x_{i+1} = x] &= \sum_{y \in S} \Pr[x_i = y] \cdot (\boldsymbol{p}_y)_x = \sum_{y \in S} \Pr[x_i = y] \cdot p_{y,x} \\
&= \sum_{y \in S} (\boldsymbol{q}^{(0)})^T \cdot \boldsymbol{P}^i)_y \cdot p_{y,x} = \sum_{y \in S} \left(\sum_{z \in S} q_z^{(0)} \cdot p_{z,y}^{(i)} \right) \cdot p_{y,x} \\
&= \sum_{z \in S} q_z^{(0)} \left(\sum_{y \in S} p_{z,y}^{(i)} \cdot p_{y,x} \right) = \sum_{z \in S} q_z^{(0)} \cdot p_{z,x}^{(i+1)} \\
&= ((\boldsymbol{q}^{(0)})^T \cdot \boldsymbol{P}^{i+1})_x \,.
\end{aligned}$$

□

Hence, linear dynamical systems can be efficiently simulated by probabilistic Turing machines, and therefore $\mathrm{P}(\mathcal{LDS}) \subseteq \mathrm{BPP}$. Since obviously also $\mathrm{BPP} \subseteq \mathrm{P}(\mathcal{LDS})$, we obtain that $\mathrm{P}(\mathcal{LDS}) = \mathrm{BPP}$.

14.2 Quadratic dynamical systems

For quadratic dynamical systems we can show the following theorem.

Theorem 14.2 *For any succinctly-defined quadratic dynamical system, the sampling problem can be solved by a probabilistic Turing machine with polynomial space.*

Proof. If we have two independent samples r and s according to the probability distribution $\boldsymbol{q}^{(t)}$, then we can generate a sample according to the probability distribution $\boldsymbol{q}^{(t+1)}$. This is true, because the probability to generate an element $\ell \in S$ after t rounds is given by

$$q_\ell^{(t+1)} = (\boldsymbol{q}^{(t)} \times \boldsymbol{q}^{(t)})_\ell = \sum_{i,j,k} q_i^{(t)} \cdot q_j^{(t)} \cdot p_{(i,j),(k,\ell)}$$

and the probability that $r = i$ and $s = j$ is exactly equal to $q_i^{(t)} \cdot q_j^{(t)}$. Hence, a naive algorithm for the generation of a sample according to probability distribution $\boldsymbol{q}^{(t)}$ can work as follows (see Figure 1 for an illustration):

Generate 2^t independent samples according to $\boldsymbol{q}^{(0)}$. Then group them into pairs and use these pairs to generate 2^{t-1} independent samples according to $\boldsymbol{q}^{(1)}$. Continue with these samples as for the previous samples, until we obtain one sample according to $\boldsymbol{q}^{(t)}$.

This naive approach can be transformed into an algorithm that only uses polynomial space:

$L_0, L_1, \ldots, L_t = \emptyset$
while $L_t = \emptyset$ do

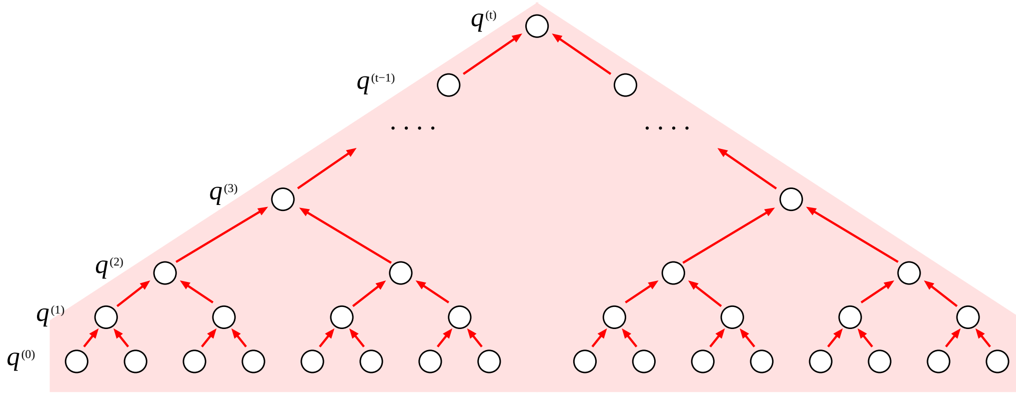


Figure 1: A naive approach to simulating a QDS.

generate a sample according to $\boldsymbol{q}^{(0)}$ and
add it to L_0
for $i = 0$ to $t-1$:
  if $|L_i| = 2$, then
    produce from the samples in L_i with $M_{\boldsymbol{P}}$ a sample and
    add it to $Li + 1$
    $L_i = \emptyset$
output L_t

This algorithm obviously needs only polynomial space. Furthermore, it can be shown that the Turing machine outputs $x^{(t)}$ with the desired probability distribution. □

Hence, $\mathrm{P}(\mathcal{QDS}) \subseteq$ PSPACE. One may ask whether this can be improved in any way. In the following we will show that this is not possible, indicating that quadratic dynamical systems are hard to simulate.

14.3 An NP-hard sampling problem

We start with a simple example. Let ϕ be an instance of SAT with n variables. We define the following quadratic dynamical system for ϕ.

Consider the quadratic dynamical system with $S = \{0, 1\}$ and a matrix $\boldsymbol{P} = (p_{(i,j),(k,\ell)})$ with entries

$i, j \backslash k, \ell$	00	01, 10	11
00	1	0	0
01, 10	0	0	1
11	0	0	1

Furthermore, choose the initial vector $\boldsymbol{q}^{(0)} = (q_0^{(0)}, q_1^{(0)})$ with

$$q_1^{(0)} = 2^{-n} \cdot |\{x\colon\ x \text{ is a satisfying assignment for } \phi\}| \,.$$

and $q_0^{(0)} = 1 - q_1^{(0)}$.

$\boldsymbol{q}^{(0)}$ is succinctly-defined, since it simply suffices to guess a random assignment and output 1 if and only if this assignment satisfies ϕ. Also $\boldsymbol{P}$ is obviously succinctly-defined. When applying $\boldsymbol{P}$ to any distribution, the following result can be shown.

Lemma 14.3 *For any distribution $\boldsymbol{q} = (q_0, q_1)$ with $q_1 \leq 1/2$, $(\boldsymbol{q} \cdot \boldsymbol{P})_1 \geq \frac{3}{2}q_1$.*

Proof. It holds that

$$\begin{aligned}(\boldsymbol{q} \cdot \boldsymbol{P})_1 &= \sum_{i,j,k \in \{0,1\}} q_i \cdot q_j \cdot p_{(i,j),(k,1)} \\ &= q_0 \cdot q_1 \cdot p_{(0,1),(1,1)} + q_1 \cdot q_0 \cdot p_{(1,0),(1,1)} + q_1 \cdot q_1 \cdot p_{(1,1),(1,1)} \\ &= 2(1-q_1)q_1 + q_1^2 = 2q_1 - q_1^2 .\end{aligned}$$

This is at least $\frac{3}{2}q_1$ for $q_1 \leq 1/2$. □

If ϕ is not satisfiable, then $q_1^{(0)} = 0$ and therefore $q_1^{(t)} = 0$ for all t. However, if ϕ is satisfiable, then $q_1^{(0)} \geq 2^{-n}$ and therefore for all t with $q_1^{(t)} \leq 1/2$, $q_1^{(t+1)} \geq \frac{3}{2}q_1^{(t)}$. In this case, it takes at most $2n+1$ rounds before $q_1^{(t)} \geq 2/3$.

These results allow us to construct a uniform family of quadratic dynamical systems for the SAT problem that, for any input ϕ, would only require $O(|\phi|)$ many steps to ensure that

- if $\phi \in SAT$, then the share of species 1 is at least 2/3, and
- if $\phi \notin SAT$, then the share of species 1 is 0.

Hence, $SAT \in \mathrm{P}(\mathcal{QDS})$.

14.4 PSPACE-hardness

Next we show that simulating an arbitrary QDS is PSPACE-hard. For this we use the PSPACE-complete problem QSAT. Recall that QSAT is defined as

$$\begin{aligned} QSAT = \{\langle\psi\rangle :\ & \psi \text{ is a valid quantified Boolean formula of the form } \exists x_1 \forall x_2 \exists x_3 \ldots Q_m x_m \phi \\ & \text{for some } m \in \mathbb{N}, \text{ where } \phi \text{ is an expression in CNF with variables } x_1, \ldots, x_m\} . \end{aligned}$$

Theorem 14.4 *$QSAT \in \mathrm{P}(\mathcal{QDS})$.*

Proof. Let ψ be an instance of QSAT. We can "unfold" ψ as a circuit of the following form (see Figure 2 for an illustration):

1. The gate consists of $n+1$ levels numbered from 0 to n. It has 2^n inputs at level n and one output at level 0. Level $i \leq n$ has 2^i gates. Every gate (except for the gates at level n) has two inputs and one output.
2. The inputs at level $n+1$ will be set to 0 or 1 in the following way: input $(b_1 b_2 \ldots b_n) \in \{0,1\}^n$ is 1 if and only if $\phi(b_1, b_2, \ldots, b_n)$ is true.
3. All gates in even levels are $\vee$-gates, and all gates in odd levels are $\wedge$-gates.

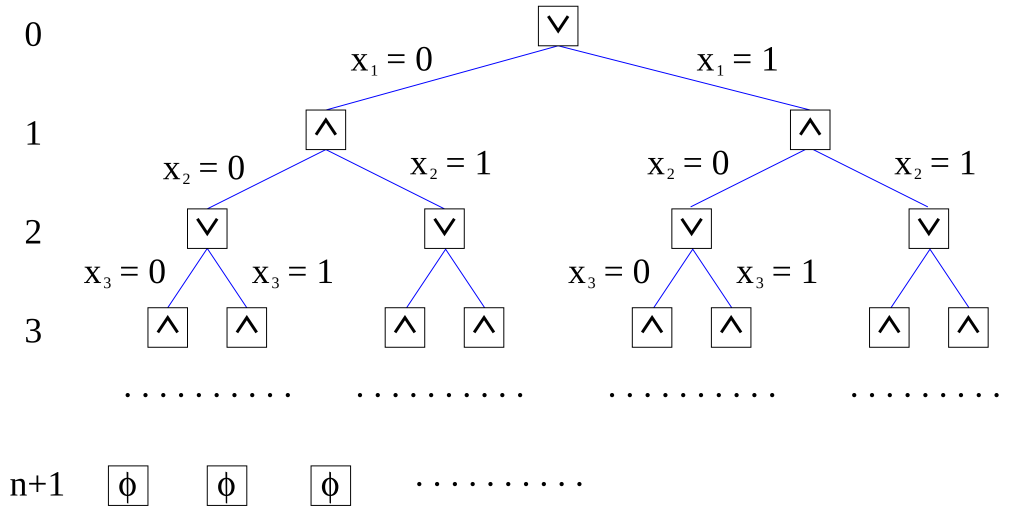


Figure 2: The layout of ψ as a circuit.

Since the circuit is nothing else than the result of applying the recursive definition given in Section 4 for a quantified Boolean expression to be valid, the circuit outputs 1 if and only if ψ is valid.

Let G be a set of all gates in the circuit above. The following QDS will simulate the operations in this circuit: We set

$$S = G \times \{0, 1\} .$$

The initial distribution $\boldsymbol{q}^{(0)}$ is the uniform distribution over the following set of size 2^n:

$$\{(g, b) : \ g \text{ is a gate and } b \text{ is its value}\} .$$

The transition matrix $\boldsymbol{P}$ is chosen in a way that all entries $p_{(i,j),(k,\ell)}$ are either 0 or 1. To simplify notation, we will therefore use in the following the notation $(i, j) \to (k, \ell)$ instead of $p_{(i,j),(k,\ell)} = 1$. The rules are:

1. $((g_1, b_1), (g_2, b_2)) \to ((g_3, b_3), (g_3, b_3))$, if gates g_1, g_2 are inputs to the gate g_3 and $b_3 = g_3(b_1, b_2)$.

2. $((g_1, b_1), (g_2, b_2)) \to ((g_1, b_1), (g_1, b_1))$, if gate g_1 is in a higher level in the circuit than g_2 (if g_2 is higher than g_1, then the result is $((g_2, b_2), (g_2, b_2))$.

3. $((g_1, b_1), (g_2, b_2)) \to ((g_1, b_1), (g_2, b_2))$, if none of the cases above is fulfilled.

It is not difficult to check that both $\boldsymbol{q}^{(0)}$ and $\boldsymbol{P}$ are succinctly-defined. The basic idea of how the QDS simulates the circuit is as follows: it starts with tupels (g, b) that represent correct assignments of values to leaves. Pairwise interaction of states according to rule 1 simulates the computation at an inner gate g_3. This is how correct values filter up the circuit. Interaction according to rule 2 boosts the probability of the gate that is closer to the output. This makes, as we will see, the probailities go up the circuit very fast. Interaction according to rule 3 leaves the states unchanged. The following lemma is immediate.

Lemma 14.5 *If at any time t, $\Pr[(g,b)] > 0$, then b is the correct value of gate g.*

Therefore, $\Pr[(g,b)]$, if it is nonzero, may unambiguously be called $\Pr[g]$.

Lemma 14.6 *If g_1, g_2 are gates at the same level, then for all time steps $t \geq 0$, $\Pr[g_1] = \Pr[g_2]$.*

Proof. The claim is true at $t = 0$. Suppose that it is true for all $t \leq k$. We will show that it is then also true for $t = k+1$. From this the lemma would follow by induction.

Note that the interactions in rules 1 through 3 treat all gates at the same level identically. Hence, if probabilities do not vary within a level at time k, rules 1 through 3 will not change probabilities within any level at time $k+1$. □

Let $p_{i,t}$ denote the probability of a level-i gate at time t. Then rule 1 implies that

$$p_{i,t+1} \geq 2p_{i+1,t}^2 . \tag{1}$$

The probability of all gates at or below level i at time t, denoted as $P_{\geq i,t}$, is

$$2^n p_{n,t} + 2^{n-1} p_{n-1,t} + \ldots + 2^i p_{i,t} .$$

Since both rules 1 and 2 move probability upwards but never downwards, $P_{\geq i,t}$ is non-increasing in t. More specifically, since a state at level $\geq i$ is produced only by mating two states from level $\geq i$, and each of these states has a probability of $P_{\geq i,t}$ of being selected, we have the following expression:

$$P_{\geq i,t+1} \leq P_{\geq i,t}^2 . \tag{2}$$

Now we are ready to prove the following lemma.

Lemma 14.7 *For any $t = 3ni + 1$ with $i \leq n$,*

$$P_{\geq n-i+1,t} \leq 2^{-n} .$$

Proof. We first prove the claim to be true for $i = 1$. Recall from the description of $\boldsymbol{q}^{(0)}$ that $P_{\geq n,0} = 1$. Then Inequality 1 implies that at $t = 1$, some of the probability moves up, so that $P_{\geq n,1} \leq 1 - 1/2^{2n}$. Now Inequality 2 implies that

$$P_{\geq n,t} \leq \left(1 - \frac{1}{2^{2n}}\right)^{2^{t-1}} ,$$

and so $P_{\geq n,3n+1} \leq \mathrm{e}^{-n}$.

Now assume the claim is true for $i \leq n+1-k$, and let $j = n+1-k$. Then

$$P_{\geq k,3jn+1} \leq 2^{-n} .$$

Since there are 2^{k-1} gates at level $k-1$, Lemma 15.6 implies that $p_{k-1,t} \leq 1/2^{k-1}$ for any time step t, and in particular for $t = 3jn+1$. Then Inequality 1 implies that in one step, some of this probability moves up to level $k-2$, and $p_{k-1,3jn+2} \leq (1 - 1/2^{k-1}) p_{k-1,3jn+1}$. Therefore,

$$P_{\geq k-1,3jn+2} \leq 1 - \frac{1}{2^{k-1}} + \frac{1}{2^n} .$$

Again, Inequality 2 shows that

$$P_{\geq k-1,t} \leq (P_{\geq k-1,t-3jn+2})^{2^{t-3jn-2}},$$

which for $t = 3(n+2-k)n+1$ gives the required conclusion. □

This means that after time $t = 3n^2+1$, the output gate has probability ≈ 1, and a random sample from $\boldsymbol{q}^{(3n^2+1)}$ enables us to determine whether or not the circuit accepts. □

Since $QSAT$ is PSPACE-complete, it follows that PSPACE $\subseteq$ P($\mathcal{QDS}$). Thus, together with Theorem 15.2, P($\mathcal{QDS}$) = PSPACE.

14.5 Simulation of general QDS by symmetric, locally reversible QDS

Next we show that even symmetric and locally reversible QDS are hard to simulate.

Theorem 14.8 *Sampling an arbitrary succinctly-defined QDS can be reduced to sampling some symmetric and locally reversible QDS with an initial distribution that is a point distribution.*

Proof. Let $\mathcal{D}$ be a succinctly-defined QDS. Given any time interval t and error parameter ϵ, we show how to transform $\mathcal{D}$ into another succinctly-defined QDS $\hat{\mathcal{D}}$ such that $\hat{\mathcal{D}}$ is symmetric and locally reversible and $||\boldsymbol{q}^{(t)} - \hat{\boldsymbol{q}}^{(t)}||_1 \leq \epsilon$.

Let S be the state space of $\boldsymbol{P}$ and let $m \geq |S|$. The state space of $\hat{\mathcal{D}}$ is $Z = \bigcup_{j=0}^{t} S \times [m]^j$, where $[m] = \{1, \ldots, m\}$. If a state $\alpha \in Z$ is an element of $S \times [m]^j$, we shall say that $level(\alpha) = j$ and write $\alpha = (\alpha_0, \ldots, \alpha_j)$. The states in $S \times [m]^j$ represent the states of the original QDS after j rounds. The reason why we use m times more states per round is to ensure that a sufficiently large share can be moved to upper levels also for symmetric and locally reversible transitions.

The transition probabilities of the new QDS are defined as follows: if $level(\alpha) \neq level(\alpha')$, then $\hat{p}_{(\alpha,\alpha'),(\alpha,\alpha')} = 1/2$ and $\hat{p}_{(\alpha,\alpha'),(\alpha',\alpha)} = 1/2$.

Now suppose that $level(\alpha) = level(\alpha') = k$. Let $\alpha_0 = s$ and $\alpha'_0 = s'$. For any pair of states $u, u' \in S$, we consider the following cases. In the following, let $p = p_{(s,s'),(u,u')}$.

The forward transitions of $\hat{\boldsymbol{P}}$ are defined as follows: Let $\mu, \mu' \in Z$ satisfy $level(\mu) = level(\mu') = k+1$, $\mu_0 = u$, $\mu_i = \alpha_i$ for all $1 \leq i \leq k$, and $\mu'_0 = u'$ and $\mu'_i = \alpha'_i$ for all $1 \leq i \leq k$. Then $\hat{p}_{(\alpha,\alpha'),(\mu,\mu')} = p(1-(|S|/m)^2)/m^2$.

If $k = 0$, then the transitions are completed by adding a self-loop of probability $(|S|/m)^2$ for each pair, i.e. $\hat{p}_{(\alpha,\alpha'),(\alpha,\alpha')} = (|S|/m)^2$.

If $k \geq 1$, there are both reverse transitions and self-loops: The probability of a reverse transition, from a pair of level $k+1$ states to a pair of level k states, is defined to be exactly equal to the probability of the corresponding forward transition (from the pair of level k states to the pair of level $k+1$ states). The self-loop for a pair of states has probability $(|S|/m)^2$ minus the sum of the probabilities of the reverse transitions from that pair.

It is clear from the definition that $\hat{\boldsymbol{P}}$ is symmetric and locally reversible. Moreover, it is not difficult to check that $\hat{\boldsymbol{P}}$ is also succinctly-defined. Since it can also be shown that $||\boldsymbol{q}^{(t)} - \hat{\boldsymbol{q}}^{(t)}||_1 \leq \epsilon$, the theorem holds. For more details, see the paper by Arora et al. □

14.6 Higher dynamical systems

In Arora et al it is shown that k-adic dynamical systems have similar properties as quadratic dynamical systems. More precisely, the following result is true.

Theorem 14.9 *Let $p(x)$ be an arbitrary polynomial. Consider an arbitrary k-adic dynamical system with $k \leq p(\log |S|)$. Then the simulation of this system is polynomially reducible to the simulation of a quadratic dynamical system.*

Thus, a high complexity theoretical jump only occurs from linear to quadratic dynamical systems.

14.7 References

- S. Arora, Y. Rabani, and U. Vazirani. Simulating quadratic dynamical systems is PSPACE-complete. In *Proc. of the ??th Symp. on Theory of Computing*, pp. 459–467, 1994.