We study the dynamics of histone modification states across multiple cell divisions.

We developed a flexible stochastic simulation system based on Gillespie’s algorithm.

We use an evolutionary algorithm to find mixtures of enzymes for stable inheritance.

Easy to evolve a system of enzymes that can stably maintain a particular chromatin state.

The difficulty of this task depends on multiple previously unanticipated factors.

Eukaryotic histones carry a diverse set of specific chemical modifications that accumulate over the life-time of a cell and have a crucial impact on the cell state in general and the transcriptional program in particular. Replication constitutes a dramatic disruption of the chromatin states that effectively amounts to partial erasure of stored information. To preserve its epigenetic state the cell reconstructs (at least part of) the histone modifications by means of processes that are still very poorly understood. A plausible hypothesis is that the different combinations of reader and writer domains in histone-modifying enzymes implement local rewriting rules that are capable of “recomputing” the desired parental modification patterns on the basis of the partial information contained in that half of the nucleosomes that predate replication.

To test whether such a mechanism is theoretically feasible, we have developed a flexible stochastic simulation system (available at http://www.bioinf.uni-leipzig.de/Software/StoChDyn) for studying the dynamics of histone modification states. The implementation is based on Gillespie’s approach, i.e., it models the master equation of a detailed chemical model. It is efficient enough to use an evolutionary algorithm to find patterns across multiple cell divisions with high accuracy.

We found that it is easy to evolve a system of enzymes that can maintain a particular chromatin state roughly stable, even without explicit boundary elements separating differentially modified chromatin domains. However, the success of this task depends on several previously unanticipated factors, such as the length of the initial state, the specific pattern that should be maintained, the time between replications, and chemical parameters such as enzymatic binding and dissociation rates. All these factors also influence the accumulation of errors in the wake of cell divisions.

© 2013 Elsevier Ltd. All rights reserved.
The terms “epigenetic inheritance” and “epigenetic memory” generally designate the capability of cells to transmit alternative gene expression programs and cellular phenotypes across cell divisions without altering the underlying DNA sequence. The carrier of this heritable information is still subject to intense research (Petruk et al., 2012). The most plausible candidates are DNA methylation (for which a faithful copying mechanism is known), histone modifications, histone variants, histone-modifying enzymes, or the general expression state of the cell. For example, it has been argued that Trithorax and Polycomb group proteins play a pivotal role in epigenetic inheritance for certain modifications (e.g., H3K4 or H3K27 methylation) and may in fact be the true epigenetic carriers rather than the histone modifications (Petruk et al., 2012). However, recent work in Drosophila indicates the opposite, namely that H3K27 (and its various modifications, particularly methylation) is indeed the crucial substrate for heterochromatin formation via Polycomb proteins (Pengelly et al., 2013).

The inheritance of histone modifications is not a trivial achievement since replication is associated with the partial replacement of histones and the deposition of newly assembled and hence unmodified histones (Annuzzato, 2005; Radman-Livaja et al., 2011). In other words, replication and the subsequent re-constitution of chromatin constitutes a dramatic disruption of the chromatin states that amounts to a partial erasure of the information stored in histone modifications.

Here, we propose that the reconstitution of local histone modification patterns is one of the biologically important computational tasks that is naturally solved by the “chromatin computer”. The need to propagate epigenetic information to subsequent generations comes in two variants. The more stringent version concerns stable heritable bistability in which epigenetic information can be transmitted for, in principle, an infinite number of generations, depending on the strength of the bistability of the underlying system. As shown by Dodd et al. (2007), this requires cooperative, positive feedback recruitment reactions as well as non-local interactions. A recent review of formal modeling approaches to the dynamics and propagation of histone modifications has been provided by Rohlf et al. (2012). In contrast, the maintenance of local patterns of histone modifications over a limited number of somatic cell divisions can potentially violate the conditions for long-term stability and tolerate slow accumulation of errors. In this setting it makes sense, therefore, to dispense with the stringent requirements outlined by Dodd et al. (2007). Indeed, it appears that this less stringent version is the relevant mechanism in multi-cellular organisms, as cells only replicate a limited number of times (the Hayflick limit). Furthermore, it has been reported repeatedly that particular epigenetic modifications can be gradually changed over generations through a number of different settings. In particular, these include the progressive reduction of higher histone methylation levels to lower methylation forms (Katan-Khaykovich and Struhl, 2005), epigenetic reprogramming (Jeong et al., 2007; Katz et al., 2009), epigenetic silencing/heterochromatin formation (Mutskov and Felsendfeld, 2003; Millar and Grunstein, 2006), and transcription-coupled histone modifications (Tippmann et al., 2012). Furthermore, histone modification gradients for a number of different modifications have been observed along a particular genomic region (for a review, see Henikoff and Shilatifard, 2011). This effect may also indirectly result from multiple cell division due to the preferential retention of parental histones at the 5′ end of genes (Radman-Livaja et al., 2011). Gradual changes of histone modification levels have been implicated to play a crucial role in aging (Przybilla et al., 2012; De Vos et al., 2011). DNA methylation changes, finally, are also intimately linked to histone modifications (Mutskov and Felsendfeld, 2003), and therefore may be a direct result of the dilution of one or more histone modifications.

One of the best-studied mechanisms proposed for epigenetic memory is based on positive feedback loops in nucleosome modification (Grunstein, 1998; Turner, 1998), suggesting that a cluster of nucleosomes may be able to stably maintain a particular state of modifications (Dodd et al., 2007; Dodd and Sneppen, 2011). Although this mechanism is not limited to silencing, histone modifications that establish and maintain repressive states (such as H3K9me3) have served as a model for this effect. Mounting evidence indicates that histone-modifying enzymes can have higher binding affinities for a particular location if neighboring nucleosomes carry the same or chemically similar modifications.
particular genomic loci (Magistri et al., 2012; Lee, 2012; Mondal et al., 2001). This may subsequently facilitate localized spreading in cis and the (re-)establishment of chromatin domains. Further contributing to the complexity of how an epigenetic memory, long non-coding RNAs seem to have crucial roles in regulating gene expression by guiding chromatin-modifying enzyme complexes to particular genomic loci (Magistri et al., 2012; Lee, 2012; Mondal et al., 2010; Khalil et al., 2009). An increasing number of other molecular pathways involving small non-coding RNAs or DNA methylation in concert with histone modifications have been described (Moazed, 2011) as well.

The coupled reading, writing, and erasing of histone modifications is therefore of crucial importance. Given the multiple tasks histone modifications are involved in, we hereafter focus on whether a simple “chromatin computer” is capable of solving the pattern completion problem for a diverse set of chromatin input states, despite the highly disruptive nature of frequent cell divisions. More to the point, we ask whether it is feasible to find combinations of reader/writer enzymes that are capable of propagating, with high accuracy, pre-set chromatin states across several cell divisions. To answer this question we implement a generic stochastic simulation of rule-based chromatin modifications as a model of the “chromatin computer”. We then employ an evolutionary algorithm to evolve “programs” representing mixtures of rewriting rules to solve various pattern reconstruction tasks.

2. Methods

2.1. A coarse-grained chemical model of chromatin computation

We begin by introducing our model of chromatin as a computational device. Computation abstractly consists of a system of states and transitions between them. Our intention is to stay close to a physical model of chromatin. Similar to much of the literature, we define a chromatin state as the set of chemical modifications of histone molecules (or their absence) located at specific genomic positions. Although this is a simplified view, the full biological complexity of the components that make up a particular chromatin state cannot be integrated at this time given that the exact underlying mechanisms are incompletely understood. Therefore, we disregard effects such as changes in nucleosome occupancy that can have an impact on gene expression (Tillo et al., 2010; Wang et al., 2011), the presence of histone variants, the effects of (local) nucleosome state patterns and cause a change in the nucleosome state pattern (Fig. 2A). They recognize parts of the nucleosome state pattern or, abbreviated, simply pattern 1^0*2^4. Such patterns therefore usually consist of multiple distinct chromatin domains (see Table 3 for examples). The chromatin string uses a modified version of the Wikimedia Commons file “Nucleosome organization.png” (licensed under the Creative Commons Attribution-Share Alike 3.0 Unported license). (For interpretation of the references to color in this figure caption, the reader is referred to the web version of this paper.)

These simplifications allow us to view chromatin as a linear sequence of nₙ nucleosomes, analogous to the well-known “beads-on-a-string” picture of chromatin. Consequently, each nucleosome (except the two boundary nucleosomes) has two immediate adjacent neighboring nucleosomes or abbreviated simply “neighbors” (see also Fig. 1). As each nucleosome is completely specified by the collection of histone modifications that it carries, it can be represented by a single character aₑₐ that encodes its modification state (nucleosome or chromatin state). The symbol 0 is reserved for the unmodified state. As a technical simplification to save memory and to expedite the recognition of patterns in practical simulations, we use a single character instead of a string to represent the state of the more than 130 known human histone modifications that have been observed (Tan et al., 2011). We define a chromatin domain as a set of adjacent nucleosomes that are in the same modification state aₑₐ. Each chromatin domain has a particular characteristic signature (e.g., methylated H4K20 and acetylated H3K9 residues) and length l and therefore can be represented by a sequence of nucleosome states 0^l. We call this a local nucleosome state pattern (or simply pattern) to emphasize that the chromatin state is solely determined by the nucleosome state. Naturally, such patterns may also be composed of multiple adjacent distinct chromatin domains (e.g., see Fig. 1 and Table 3).

Transitions between chromatin states are mediated by histone-modifying enzymes. These enzymes catalyze the writing or erasing of histone modifications in a context-dependent manner. Therefore, they are implemented as string rewriting rules acting on the nucleosome state pattern (Fig. 2A). They recognize parts of the (local) nucleosome state patterns and cause a change in the modification state of one or more nucleosomes. These rewriting rules are described in detail in the following section.

Not all chromatin-modifying enzymes are present in the cell at the same time or in same concentrations, and they may differ substantially in their affinity to their target patterns or in their catalytic efficiency. Furthermore, different enzymes may compete for the same chromatin locations and, vice versa, different chromatin positions may compete for low abundance enzymes (Fig. 2C). We therefore model the dynamics of the “rewriting reactions” in a way that follows the mass action kinetics of chemical reaction systems, distinguishing, for each enzyme and each chromatin position, their elementary reactions (Fig. 2B):

1. Binding of an enzyme to a specific locus (i.e., one or more nucleosomes). We assume that enzymes only bind to nucleosomes on the chromatin string that match the pre-condition of
the rewriting rule that they embody. Also, enzymes cannot bind if any of the nucleosomes that are decisive for the applicability of the rewriting rule are bound by other enzymes, thereby blocking the accessibility.

2. Dissociation of an enzyme and the application of a rewriting rule (i.e., writing or erasing of one or multiple histone modifications). After dissociating, the enzyme is again available for new reactions.

Contrary to most previous work and simulation systems for chromatin state dynamics, the computational paradigm used here for the enzyme kinetics is a stochastic one. We argue that this level of chemical realism is crucial, since concentrations of regulatory molecules, rather than their mere presence or absence, are very well known to be of crucial importance in the regulation of gene expression. Indeed, few regulatory events are qualitative—typically changes in expression levels of regulators are gradual and rarely exceeding a few-fold increases or decreases.

Chromatin state dynamics are thus dependent on enzyme abundances, the availability of local patterns on which they can act, the current state of the system, and the rate constants for each chemical reaction (which generally quantify the speed of a chemical reaction and may differ substantially among different enzymes) (Fig. 2C).

Histone modifications have strikingly different lifetimes and are deposited at different rates. Acetylation events are measured in the order of minutes, while methylation events are stable for days (Barth and Imhof, 2010). These rate differences are determined by the enzymes that catalyze the corresponding reactions (Hathaway et al., 2012). A given mark can be removed either by specific de-modification enzymes or through chromatin remodeling (e.g., histone turnover or histone tail clipping). Since chromatin remodeling phenomena are at present not explicitly included in our model, different life-times can be modelled by neighbor-independent rewriting rules with different rate constants. To our knowledge, spontaneous (i.e., enzyme-independent) decay has not yet been described for histone modifications, although it cannot be excluded that some of the more exotic or yet undescribed modifications may not require an enzyme for de-modification.

For simplicity, we use a single reaction rate parameter for the binding of histone-modifying enzymes or enzyme complexes, although mechanistically, this may require multiple steps (e.g., binding, recruitment of other factors and oligomerization). The propensity for a particular binding reaction is computed as the product of its reaction rate and the number of free (i.e., not bound)
molecules for that enzyme, while for any dissociation reaction, the propensity equals its reaction rate and is therefore independent of the number of free molecules.

The time course of the simulation between replications can be subdivided into discrete “phases” (Fig. 3) that can have varying durations. For each phase, enzymes can be arbitrarily set to be present or absent (which will subsequently call enzyme availability) (Fig. 3). However, enzyme concentrations either have present or absent (which we will subsequently call enzyme availability) (Fig. 3). Each state can be interconverted by (i) a constant value as specified by the "chromatin computer" programming (present), or a value of 0 (absent). If an enzyme is still bound at the transition from one phase to the next, it dissociates without performing a state change at the nucleosome it binds to. Collectively, these phases abstract the cell’s gene expression program. We may interpret them for instance as the G1, S, and G2 phase of the cell cycle. Alternatively, phases may be used to model distinct developmental stages.

We also include replication in our model. At regular time intervals, a replication takes place and the parental histones are distributed between the two daughter strands. Several models for histone segregation and deposition have been proposed in the last years, and due to the crucial importance of replication for any chromatin-based inheritance, a few of them will be presented in some detail in the following. The random or conservative model assumes that parental histones are deposited randomly between both strands while the gaps are filled with newly assembled and unmodified nucleosomes (Jackson and Chalkley, 1985; Corpet and Almouzni, 2009; Hansen et al., 2008). Thus, at any position in each of the strands, the probability that parental histones are incorporated, but histone-modifying enzymes remain bound during replication or quickly rebind to restore the parental modification pattern.

The model with the best experimental support is a random model in which nucleosomes disassemble into one (H3–H4)2 tetramer and two H2A–H2B dimers that are randomly deposited (Annunziato, 2005; Zhu and Reinberg, 2011; Radman-Livaja et al., 2011). All other models are either purely hypothetical or only weakly supported. For example, a truly semi-conservative replication would require that, within one nucleosome, histones of the same type carry identical histone modifications, a condition that however is not always fulfilled (Voigt et al., 2012; Tran et al., 2012).

For the purpose of the present study, we therefore adopted a random model (Fig. 2). We treat nucleosomes as indivisible units, which however is only a simplification if multiple histone modifications from different histones are modeled. We note, however, that the simulation environment can easily be extended to other replication models if the need arises. Again, we argue that this additional level of biological realism is irrelevant for the questions addressed in this contribution. We also assume that parental histones are re-deposited at their pre-replication locus, which is consistent with the finding that most parental histones in budding yeast are re-incorporated in close vicinity, i.e. within 400 bp, of their original locus (Radman-Livaja et al., 2011). Lastly, analogous to the phase change transitions, enzymes that are still bound at the time of replication dissociate without performing any reaction.

2.2. Chromatin enzymes as rewriting rules

In the simplest case, the enzymes evaluate the state of the nucleosome that they modify. The model of Dodd et al. (2007), for example, considers three distinct states: unmodified (0), methylated (m), and acetylated (a). Each state can be interconverted by the catalytic actions of histone acetyltransferases (HATs), histone deacetylases (HDACs), histone methyltransferases (HMTs), and histone demethylases (HDMs). The corresponding set of rewriting rules is

\[
\begin{align*}
\text{HAT:} & \quad 0 \rightarrow A \\
\text{HDAC:} & \quad A \rightarrow 0 \\
\text{HMT:} & \quad 0 \rightarrow M \\
\text{HDM:} & \quad M \rightarrow 0
\end{align*}
\]

(1)

As opposed to these simple, neighbor-independent rewriting rules, more complex ones, such as the ones considered in Sneppen and Dodd (2012) or Bryant (2012), also depend on the neighboring nucleosomes (neighbor-dependent rewriting rules). In our implementation, arbitrarily complex rewriting rules can be specified (see Table 1 and Fig. 2 for a few examples). For convenience, we allow wildcards in the definition of the rewriting rule and the state of the neighboring nucleosomes can be incorporated as well. Rules are interpreted as symmetric with respect to their left and right neighbors as it is not possible for the enzyme complexes to determine directionality (e.g. towards the centromere) from the local chromatin structure. In reality, it appears that asymmetries

<table>
<thead>
<tr>
<th>Rewriting rule</th>
<th>Matching patterns for the rewriting rule</th>
<th>Neighbor-dependent</th>
</tr>
</thead>
<tbody>
<tr>
<td>[1]0</td>
<td>[1]0</td>
<td>No</td>
</tr>
<tr>
<td>[1,2]0</td>
<td>[1,2]0</td>
<td>No</td>
</tr>
<tr>
<td>[1]0</td>
<td>[1]0</td>
<td>Yes</td>
</tr>
<tr>
<td>[1]0</td>
<td>[1,2]0, [1]0</td>
<td>Yes</td>
</tr>
<tr>
<td>[1]0</td>
<td>[1,2]10</td>
<td>[1]0, [1,2]10</td>
</tr>
<tr>
<td>[1]0</td>
<td>[1,2]10, [1]0</td>
<td>[1,2]10</td>
</tr>
<tr>
<td>[1]0</td>
<td>[1]0, [1,2]10, [1,2]10, [1]0</td>
<td>[1,2]10</td>
</tr>
</tbody>
</table>
are induced locally and thus are oriented relative to features such as promoters, insulators, or transcription factor binding sites (Kundaje et al., 2012).

Such complex rewriting rules indeed seem to be common for eukaryotic systems, as histone-modifying enzymes are often part of large enzyme complexes with multiple protein domains, each of which having a particular function such as DNA sequence recognition or histone binding. Individual histone modifications in the vicinity of the binding may substantially alter the binding affinity of the enzyme complex or its formation. This crosstalk has been repeatedly demonstrated within histones, among histones of the same nucleosome (Wang and Patel, 2011; Lindroth et al., 2004; Fuchs et al., 2011; Ruthenburg et al., 2011), and among histones of neighboring nucleosomes (for an overview, see Musselman et al., 2012). Even for single domain proteins, the presence of multiple modifications may be required for binding (Morinère et al., 2009).

We represent these complex rules as follows: Each rule is specified by a pair of strings, the left part is the template that must be matched to the local nucleosome state pattern. This precondition must be met so that the rewriting rule can be applied to the matched string. Once a match is found, the left string is replaced by the right string. In our implementation, both strings consist of up to three parts: (i) a string of length \( l_1 \geq 0 \) for the required state \( s \) of the neighboring nucleosome(s) left of the actual binding site of the corresponding enzyme; (ii) a string of length \( l_2 > 0 \) for the required state(s) of the nucleosome(s) that the enzyme binds to; (iii) a string of length \( l_3 \geq 0 \) for the required state(s) of the neighboring nucleosome(s) right of the binding site. Only the nucleosomes of the binding part may be subject to change in the right part of the rewriting rule (as indicated by \( a^n \) in the equation below). Thus, rules are not reversible per se; instead, reversible reactions must be formulated as separate forward and backward reactions. Furthermore, each rewriting rule is associated with a rate constant. In summary, it is formally specified in the following form:

\[
a_1 \ldots a_{i-1}[a_{i+1} \ldots a_{i+l_1}]a_{i+l_1} \ldots a_{i+l_2} \ldots a_{i+l_3} \xrightarrow{\text{rate}} a_1 \ldots a_i[a_{i+l_1} \ldots a_{i+l_2}]a_{i+l_2} \ldots a_{i+l_1} + a_{i+l_3} + a_i + \tau
\]

where each \( a \in A \) denotes a particular nucleosome state. Table 4 compiles the rewriting rules with \( l_1, l_2, l_3 \leq 1 \) that are used throughout this contribution.

The nucleosome string can be either linear or circular. For the former, rewriting rules that require the state of both neighboring nucleosomes cannot match the two boundary nucleosomes. This entails that they keep their parental status unless replication replaces them with unmodified nucleosomes, after which their original state is lost permanently (see Fig. 5). In the latter case, the two boundary nucleosomes are directly connected, and rewriting rules may match.

### 2.3. Stochastic simulation using the Gillespie algorithm

Chromatin state dynamics are modeled using Gillespie’s well-established stochastic simulation algorithm (SSA) (Gillespie, 1976, 1977). This approach reproduces the master equation of the underlying stochastic model with a discrete number of objects and continuous time. The algorithm is rejection-less and hence efficient in terms of the simulation, but it requires that possible “reaction channels” (i.e., the transitions between two particular states) are enumerated and finite. Each reaction channel is weighted with its reaction rate, which is crucial for exactness. In essence, given an initial time \( t_0 \) and population state vectors (here: enzyme concentrations), the algorithm works as follows: In each iteration, two random numbers \( r_1 \) and \( r_2 \) are drawn from the uniform distribution in the unit interval, which are subsequently used to determine the value of the time increment \( \tau \) and to select a reaction channel, respectively. Importantly, time increments are not fixed but are of variable length, depending on the current state of the system. The selected reaction channel is then executed and the state of the system changes accordingly. This process is repeated until a stop criterion has been reached (e.g., a specified simulation time). As this algorithm is well established, we refer to Gillespie (1976, 1977) for more details.

Each replication event occurs periodically after a fixed time interval \( t_r \). Similarly, each phase \( p_i \) has a defined duration and stops after a particular time \( t_p \). This requires a correction for the last reaction event in each period—i.e., the one for which \( t + \tau > t \), and \( t + \tau > t_n \), respectively. Here, we draw a random number \( r_3 \) and accept the reaction if \( r_3 > (t_n - t)/\tau \) and \( r_3 > (t_p - t)/\tau \), respectively.

Since the set of available reaction channels and their weights are, in general, dependent on the current state of the system, the “bookkeeping” of all reaction channels and their status is an important issue for the practical implementation of the Gillespie algorithm. This is particularly relevant for models with large numbers of different molecules and reactions. We next describe the peculiarities and design decisions that are specific to our chromatin-based model. To do so, we make use of abbreviations for relevant parameters that are summarized in Table 2. Consider a system with \( n_e \) enzymes, each of which has a particular number of rewriting rules. The sum of all rewriting rules is then \( n_{\text{rules}} = \sum_{k=1}^{n_e} |e_k| \), where \( |e_k| \) denotes the number of rewriting rules that are defined for that particular

<table>
<thead>
<tr>
<th>Par.</th>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Specific to the evolutionary algorithm</td>
<td>( n_e )</td>
<td>1000</td>
</tr>
<tr>
<td></td>
<td>( n_s )</td>
<td>10</td>
</tr>
<tr>
<td></td>
<td>( n_p )</td>
<td>1–4</td>
</tr>
<tr>
<td>Specific to the biology and the stochastic simulation</td>
<td>( n_r )</td>
<td>50</td>
</tr>
<tr>
<td></td>
<td>( n_n )</td>
<td>30–150</td>
</tr>
<tr>
<td></td>
<td>( c )</td>
<td>Circular</td>
</tr>
<tr>
<td></td>
<td>( t_r )</td>
<td>20</td>
</tr>
<tr>
<td></td>
<td>( t_p )</td>
<td>20</td>
</tr>
<tr>
<td></td>
<td>( n_a )</td>
<td>5 or 10</td>
</tr>
<tr>
<td></td>
<td>( n_b )</td>
<td>1–5</td>
</tr>
<tr>
<td></td>
<td>( k )</td>
<td>30</td>
</tr>
<tr>
<td></td>
<td>( n_m )</td>
<td>10</td>
</tr>
<tr>
<td></td>
<td>( n_e )</td>
<td>( n_b \times n_m )</td>
</tr>
<tr>
<td></td>
<td>( r_b )</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>( r_d )</td>
<td>5</td>
</tr>
</tbody>
</table>

Table 2: Summary of the most relevant parameters that we used for the evolutionary algorithm. See text for details.
enzyme. In our system, each rewriting rule adds a total of $2n_n$ reaction channels (one corresponding to the binding and one to the dissociation for each of the $n_n$ nucleosomes). Thus, the total number of reaction channels is $2n_n \times n_{rules}$. Equivalently, each of the $n_n$ nucleosomes has $2n_{rules}$ different reaction channels.

Internally, each rewriting rule is associated with a state change vector, which describes how the concentrations of the available $n_i$ enzymes are affected upon execution of the reaction that specifies the rewriting rule. For example, any binding reaction decreases the concentration of the corresponding enzyme by 1, whereas any dissociation reaction frees the enzyme and thereby increases its concentration by 1.

To save computing time, only the propensities of the reactions in the vicinity of the nucleosome(s) subject to the last reaction are re-computed. A recomputation of the reaction propensities is also necessary after each replication and phase transition.

### 2.4. Evolutionary optimization of the rewriting rule sets

Our “chromatin computer” may operate with rather complex instructions that correspond to a particular gene expression pattern. More formally, an instruction consists of a list $L$ of rewriting rules and associated enzyme concentrations (see Fig. 3). This is similar to the computational model of Bryant (2012), except that we have incorporated a concentration associated with each rewriting rule that modulates the probability with which it is applied. If the time course of the simulation is divided into $n_p$ phases, more complex programs can be implemented as sequences $(L_i, \tau_i), i = 1, \ldots, n_p$, of instructions that are valid for a prescribed time period $\tau_i$ (i.e., a “phase”) before being supplanted by the next instruction. Thus, if a particular rewriting rule $r_k \in L_i$, the enzyme performing this reaction is available in phase $i$.

Following previous studies that used evolutionary optimization, we define mutation and recombination operators for the individual instructions as follows:

- The concentration of a random enzyme can be decreased by up to two molecules. The concentration of a different random enzyme is correspondingly increased to keep the total number of enzymes constant.
- We can replace up to two enzymes and their corresponding rewriting rules by randomly picked different enzymes. The concentration variables and rate constants remain unchanged.
- In principle, the rate constants can be changed. However, this option has been disabled in the simulations, as discussed below.
- The recombination (cross-over) operator builds a convex combination of two instructions $L_1$ and $L_2$ using the formula $\xi L_1 + (1-\xi)L_2$ with a randomly drawn weight $\xi \in [0, 1]$.
- A second cross-over operator may similarly construct a combination of two already present enzymes and their associated rules, therefore adding a new enzyme in a non-random fashion. Due to the relative simplicity of the rules used in simulations, however, such operators have not been used here.
- If multiple instructions are used, their individual durations $\tau_j$ can be changed. For this, two randomly picked phases increase and decrease their individual durations by 10%, respectively (with respect to the time $\tau_j$ between two replications). In addition, independently for each phase, enzyme availabilities (presence or absence, see Fig. 3) of up to two random enzymes can be inverted.

The fitness of an instruction (or a schedule of instructions) is evaluated by comparing the patterns immediately before each of the cell divisions to the start pattern using the normalized Hamming distance. The initial patterns used here are compiled in Table 3. This yields $r + 1$ distance values $d(i)$. For each start pattern and $n_i$ independent starting point, we run the Gillespie simulation $n_g$ times with different random number seeds. Then, we average the distance values over the Gillespie realizations, obtaining $1 – \langle d(i) \rangle$ as the autocorrelation function of the pattern. The fitness value is next computed as the sum of this autocorrelation function over the $n_i$ cell divisions. It turns out that a simple hill-climbing approach is sufficient to obtain good solutions. Hence, a proposed mutation of the instruction (or schedule of instructions) is accepted if the estimated fitness increases. We stop the search if the best solution among all runs does not improve for $n_i$ iterations.

#### 2.5. Simulations

For all patterns, we generated $n_i$ initial instructions by randomly selecting $n_i$ out of a total of 28 rewriting rules (i.e., the enzymes that implement them) as listed in Table 4. The individual concentrations of these $n_i$ enzymes were also assigned randomly so that the total number of molecules equaled $n_m$. All other parameters were kept constant between the independent runs. Table 2 summarizes the parameters that were used for the evolutionary algorithm and the individual stochastic simulations required for the fitness evaluation.

We simulated local chromatin state dynamics for a genomic region of 12 kb to 30 kb (depending on the pattern), a range that was also used in previous approaches (Dodd et al., 2007) and that provides a good balance between computational speed and biological verisimilitude. We used a value of 50 for $n_m$, as this reflects the Hayflick limit for how many times a cell can divide. Also, we used a circular nucleosome organization throughout the simulations to avoid artifacts for the boundary nucleosomes due to the rewriting rules and their neighbor-dependence. For a suitable number of enzymes available for the modeled chromatin region, we chose a value of 10 per chromatin domain (parameter $n_m$). Dependence on the number of chromatin domains is necessary to ensure comparability among patterns with varying length due to the concentration dependence of the enzyme reactions in the Gillespie algorithm. For the elementary patterns, up to five different enzymes may be selected, whereas we increased $n_m$ to 10 for composite
patterns. We found these values to be sufficient to obtain good solutions in our simulations. The values for \( t_s \), \( r_s \) and \( r_p \) were chosen so that (i) enough reactions can take place between two subsequent replications, and (ii) the system tends to have free molecules available rather than a condition where all molecules are bound.

The unmodified state behaves differently from modifications since it is not distinguishable from erased information after replication. It may be helpful to allow more than one phase to obtain good solutions. Coordinated, phase-dependent enzyme availabilities may make it easier to systematically recompute the parental modification state. To identify the optimal number of phases \( n_p \), we first ran the evolutionary algorithm for the pattern \( 01 \) for solutions with one, two, three, and four phases and identified the solution with the highest score. We then ran all composite patterns that contain the unmodified state with only \( n_p \) phases (instead of variants with one, two, three, or four phases, respectively).

### 3. Results

Using the flexible software system that we developed (see end of results for details) to study the dynamics of histone modification states, the difficulty of pattern reconstruction problem depends on the structure of the start pattern. Therefore, we initially summarize the observations for simple, elementary start patterns (see Tables 5, 6 and Fig. 4). Stable solutions for constant patterns and patterns that consist of only modified nucleosomes were achieved with relative ease. Neighbor-independent rewriting rules were only rarely selected (or only in low concentrations), as they easily introduce noise to the system. Similar to the results of Dodd and Sneppen (2011) and Hodges and Crabtree (2012), we found that chromatin domains can transiently multifurcate to form multiple smaller domains that remain stable for a particular amount of time, which was particularly pronounced for patterns that contain patches of unmodified nucleosomes (Fig. 5). Noteworthy, we sometimes observed a gradual accumulation of errors during the lifetime of a cell (e.g., see the least stable solution for pattern \( 12 \) in Fig. 4).

**Constant patterns:** As expected, it is trivial to find optimal solutions for the constant pattern \( 1 \) as the only rewriting rules required to recompute the parental pattern are either \( a_1 (0) \rightarrow [1] \) or \( a_2 (0) \rightarrow [0] \) or \( 1 \rightarrow \). These must be present in higher concentrations than rewriting rules that change \( 1 \) to \( 0 \) (class \( \gamma \)). Consequently, a large number of simulations achieved the optimal score. Also, contrary to other patterns, the inclusion of neighbor-independent rewriting rules, such as \( a_1 \), pose no disadvantage to the system.
Pattern 12. We also found that it is relatively easy to evolve a system that can stably maintain patterns when the parental nucleosome state consists of several chromatin domains of modified nucleosomes, as exemplified by patterns 12 and 121. Four rewriting rules with approximately equal concentrations were sufficient for stable inheritance over 50 generations: \( \alpha_3 \) \{1[0][1 → 1][11], \( \alpha_5 \) \{0[0][1 → 0][11] or 1[0][0 → 1][10], \( \beta_4 \) \{2[0][2 → 2][2][2], and \( \beta_6 \) \{0[0][2 → 0][2][2] or 2[0][0 → 2][2][0].\ Notably, the boundary between differentially modified regions was subject to stochastic fluctuation given that it was controlled solely by the available rewriting rules (Fig. 4).

Pattern 01. It is substantially more difficult to find good solutions for patterns that contain a mixture of modified nucleosomes and unmodified nucleosomes such as 01, 101, or 102. The intuitive reason is that the 0-state in the target pattern is indistinguishable from the information lost during the replication event. Thus, the chromatin computer’s programs lack means to determine where states need to be regenerated and where the target state has already been reached. No good solutions seem to exist for this problem when only a single phase is allowed between replication events. To illustrate this, consider the pattern 0\( k \)1.\k.\ Rewriting rules are needed to re-establish the right part of the pattern that contains the nucleosomes in state 1 after each cell division. These can principally be constructed in two ways: (i) \ The rewriting rule \( \alpha_3 \) \{1[0][1 → 1][11]\ ensures that it can only be applied in the right part. However, having only this rewriting rule is too strict, as patterns such as 01001 cannot be repaired otherwise. This would be possible with (ii) \( \alpha_5 \) \{0[0][1 → 0][11] or 1[0][0 → 1][10],\ however this is also applicable at the boundary of the 0 and 1 region and can therefore slowly spread into the 0-region, leading to gradual loss of the ancestral signal. Rewriting rules that do not incorporate the state of neighboring nucleosomes (e.g., \( \alpha_5 \) \{0[0][1 → 1][11]\}) are also not helpful in this regard, since they introduce additional noise to the system.

Admitting more phases, i.e., variation of the gene expression program through the cell cycle, can mitigate this difficulty. Different phases can then serve a particular purpose and, collectively, they aim at restoring the parental pattern. In practice, we found that solutions with more than one phase become only marginally better with two or more phases, possibly due to the greatly expanded parameter space of the solution (Table 5). It may therefore be necessary to run the evolutionary algorithm for a substantially longer time to obtain good solutions that consist of more than one phase. Nevertheless, the stability of the best solutions we found (independent of the number of phases) is not comparable to solutions for patterns 1 and 12. This is indicated by both the attainable scores and by the time course visualization of the patterns in Fig. 4.

Composite patterns: For the composite patterns, the results were as follows: Patterns that are combinations of all three elementary patterns (i.e., 0–1, 0–2, and 1–2 chromatin domain transitions) produced the lowest scores, despite allowing the evolutionary algorithm to increase the maximal number of active enzymes (see below). The best scores were produced by variations of the pattern 12. For the pattern 12021 and 1011, the 0 patch in the middle of the patterns was consistently lost, even in the best solutions.

Fig. 4. Results from the evolutionary algorithm for the three elementary patterns (top: 1, middle: 12, bottom: 01, 2 phases). The leftmost figure in each row shows the evolution of the best score (separately for each of the \( n \) independent starting points). The second leftmost figure shows the evolution of the score of the solution that was proposed in each particular iteration. The two rightmost figures in each row display the stochasticity of the best solution among all \( n \) independent stochastic simulations (left: most stable (highest score), right: least stable (lowest score)). For each visualization, the state of the system is shown directly before each replication (with the initial state at the top, and the state after the last replication at the bottom). The coloring of the different nucleosome states is analogous to the previous Figures (0: white, 1: red, 2: blue). For pattern 1, no variation was detectable between the best and the worst performing run when states were only compared directly before each replication event. (For interpretation of the references to color in this figure caption, the reader is referred to the web version of this paper.)
We also found that solutions for the three elementary patterns can be used with comparable quality for repetitions of elementary patterns. A good example is the pattern 12, where all of the variations produced produced scores that differed by less than 10% from the original score. This occurred even though the patterns were either more complex (121, 1212) or contained chromatin domains of unequal length (122, 1222). For the pattern 01, we observed a similar result with the patterns 010, 011, 101, 0101, and 1011, although the scores were up to 20% smaller (data not shown). In all cases, the score may be improved by adjusting enzyme concentrations and phase durations, particularly for patterns with a different length than the original pattern.

However, we found that individual solutions from elementary patterns cannot merely be combined for a more complex pattern, particularly with multiple phases. For example, the combination of the individual solutions for the patterns 01, 02, and 12 does not produce a good result for patterns that contain all these three types of transitions such as 102. The combination of different solutions—i.e., the simultaneous presence of more enzymes—apparently interferes with the “strategy” of the partial solutions. The additional enzymes act at the newly produced modifications and obliterate the nascent pattern.

We also investigated whether the performance could be improved by optimizing the rate constants for binding and/or dissociation rates, along with the rewriting rules themselves. Although the scores were comparable, we obtained solutions that are much more tailored towards the reconstruction of a particular initial pattern length (data not shown). This is because the reaction rate constants also control how many reactions may take place during a particular time in the stochastic simulation. By allowing them to vary, the enzymes and the corresponding reaction rates become tailored to the specific pattern length. Neighbor-independent enzymes were then also more frequently selected, because their frequency of selection can be controlled by the reaction rates.

Finally, we examined to what extent the parameter $n_a$, which limits the maximal number of active enzymes, has an influence on the quality of the solutions. We found that selection sometimes tends to increase the number of enzymes by including rewriting rules that are rarely applicable. This was particularly true for the pattern 01, where the score for the best solution with one phase was increased to that of solutions with multiple phases using the original value of $n_a$ (data not shown). This appears to be a means of adjusting reaction rates to decrease the number of reactions that take place between replications.

To verify that the best solutions for the different patterns are not specific to the parameterization of the model and the specific pattern length, we also tested the sensitivity to parameter variations (Table 7). Specifically, we tested the effect of a linear nucleosome string rather than a circular one, the time between two replications, the number of nucleosomes and replications, and the dissociation rate of all enzymes. In summary, we found that the solutions produce very similar scores in most of the parameter space for the patterns 1 and 12, while for the pattern 01 and its compositions, the best solutions strongly depend on the kinetic parameters. We also found that the number of nucleosomes must not be too low (a value around 40 was sufficient for robustness). Otherwise, stochastic effects may irreversibly destroy the parental signal. Similarly, the time $t$, between two replications must be long enough to allow for recombination of the parental pattern. Some noteworthy effects that we observed while varying the parameters are summarized in Fig. 5.

The source code of a C implementation of our software system can be obtained under the GNU Public License from http://www.bioinf.uni-leipzig.de/Software/StoChDyn and consists of two separate programs: the stochastic simulation of histone modification dynamics using Gillespie’s approach (StoChDyn) and the evolutionary algorithm (Evo-ES) that uses StoChDyn to evaluate its solutions.

4. Discussion

In this work, we have queried whether the propagation of histone modification patterns across cell divisions can be seen as a computational problem and if so, whether chromatin is organized in a way that is amenable to the solution of this problem. Our answer is twice affirmative. We demonstrated that the faithful propagation of patterns of histone modification can be interpreted as a computational problem that is achievable through a small collection of rewriting rules. These rewriting rules are abstractions of a well-described class of enzymes and enzyme complexes combining reader, writer, and eraser domains for specific histone modifications. For the best solutions, the evolutionary algorithm selected almost exclusively enzymes that are dependent on the states of neighboring nucleosomes (except for the trivial pattern 01). This highlights that context-dependency is crucial for such inheritance systems, as context-independent enzymes easily introduce too much noise to the system that further complicates the recombination of parental state patterns after cell division. Indeed, for many histone-modifying enzymes, it is well-known that their binding affinities are highly influenced by the presence or absence of particular histone modifications or other signals nearby.

This modification process is intrinsically stochastic and crucially depends on concentrations of the available enzymes and histone modifications. Hence, we propose a cellular automata-like 1-D string as the computational paradigm for a chromatin computer, on which sets of local rewriting rules are applied asynchronously with time-dependent probabilities. As a practical implementation, we employ a detailed stochastic simulation of chromatin state dynamics to approximate the physico-chemical constraints of our approach.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Original value</th>
<th>New, varied values</th>
<th>Pattern</th>
</tr>
</thead>
<tbody>
<tr>
<td>Nucleosome organization ($c$)</td>
<td>Circular</td>
<td>Linear</td>
<td>All</td>
</tr>
<tr>
<td>Number of nucleosomes ($n_0$)</td>
<td>60</td>
<td>10–200</td>
<td>All</td>
</tr>
<tr>
<td>Number of replications ($n_r$)</td>
<td>50</td>
<td>5–100</td>
<td>$s &gt; 5$</td>
</tr>
<tr>
<td>Time between replications ($t_r$)</td>
<td>20</td>
<td>0.5–40</td>
<td>$s &gt; 5$</td>
</tr>
<tr>
<td>Dissociation rate ($r_d$)</td>
<td>5</td>
<td>0.05–100</td>
<td>$s &gt; 5$</td>
</tr>
</tbody>
</table>

Table 7

Summary of the robustness analyses for the best solution for each elementary pattern. The table lists the parameter value intervals when the score of the modified solution (with one changed parameter) achieved more than 90% of the score of the original best solution. The variable $s$ denotes the step size. When the number of replication $n_r$ was varied, we calculated the ratio of the score and the corresponding maximal possible score and compared the two ratios using the 90% threshold for better comparability, as $n_r$ has a direct influence on the maximal possible score.
Fig. 5. Visualizations from selected simulations for pattern 12 of the robustness analysis. One distinct parameter is fluctuated in each row (A–E, as indicated on the left) as compared to the best solution for this pattern, and two exemplary runs of the same simulation (except for E) are shown that highlight some noteworthy effects that were observed. The coloring is analogous to Fig. 4. (A) Variations in the chromatin organization at the boundaries (here: linear instead of circular chromatin string). (B) Variations in the number of nucleosomes (here: 20 instead of 60), illustrating that reducing the number of nucleosomes increases the likelihood that the signal is lost due to stochasticity. (C) Variations in the number of replications (here: 100 instead of 50), showing that chromatin domains can gradually disappear (left) or change their exact location due to stochasticity (right). (D) Variations in the time between replications (here: 0.5 instead of 20), highlighting that the original signal may gradually get lost (either fully or partly) if the cell has not enough time to recompute the parental pattern. (E) Variations in the dissociation rate of the enzymes (here: 0.05 (left) and 0.15 (right), respectively, instead of 1), illustrating that the enzyme kinetics may also have a large effect on pattern stability. The binding rate constants were unmodified; however, the dissociation rate constants have been reduced, which means that enzymes need more time to perform their designated reaction. (For interpretation of the references to color in this figure caption, the reader is referred to the web version of this paper.)
Here, the probability for applying rewriting rules is modeled explicitly in reaction rates for binding and dissociation following the laws of mass action. The “software” for this type of chromatin computer is thus a sequence of sets of rewriting rules and concentrations that can be directly interpreted as part of the cell's gene expression program.

We have shown that stable propagation of complex patterns without the need for explicit boundary elements is possible in our model, although not for all types of patterns. This is consistent with the findings of Hodges and Crabtree (2012), who recently proposed that explicit boundary elements may not be required for H3K9me3 domains, as their size and propagation may be naturally limited by chromatin remodeling processes. Nevertheless, the maintenance of (approximate) boundaries between differentially modified chromatin domains is a main challenge of any epigenetic inheritance mechanism. In practice, additional security measures that prevent spreading may furthermore be required for faithful long-term inheritance. Indeed, several mechanisms that prevent spreading may trivially solve the problem of restoring the parental modification pattern when exact boundaries are important. Examples include nucleosome-depleted regions (see, for example, Jang and Pugh, 2009; Mavrich et al., 2008; Yadon et al., 2010), bound proteins (e.g., CTCF) or histone variants (e.g., H2A.Z) in the vicinity of positions that border the different chromatin domains, or the marking with a particular histone modification directly after replication due to other signalling factors.

In our model, long-range interactions are not necessarily required for stability, which contrasts to what is argued by Dodd et al. (2007). However, the presence or absence of long-range interactions has no impact on the computational power of the chromatin computer or our conclusions with respect to epigenetic inheritance. Generally, epigenetic inheritance should be seen as an ensemble of different strategies that collectively aim to transmit a particular chromatin state throughout cell division. Histone modification patterns fit this bill. The propagation of one out of several alternative modified states provides the required memory across divisions. Nevertheless, only a few primary histone modifications that typically form large and homogeneous chromatin domains (e.g., H3K9me3) may be copied in a self-propagating manner as described here. The transmission or recomputation of other, more secondary histone modifications, however, likely depends on other factors (Sarkies and Sale, 2011). On the other hand, the inheritance of promoter-specific modifications (which typically cover only a few nucleosomes) is likely implemented in a different way than the propagation of large homogeneous chromatin domains, because short domains are much more difficult to inherit due to pure stochasticity (e.g., see Fig. 5B).

Pattern stability is influenced by a number of factors, such as dynamic chromatin remodeling events, the up- and downregulation of genes that code for or regulate the corresponding histone-modifying enzymes, pattern complexity (Sneppen and Dodd, 2012), or the length of the state to be maintained. In summary, histone modification patterns are often an ongoing enzymatic rewiring process that must be maintained by other genes that alter their state. The propagation of one out of several alternative modified states is therefore a way to achieve a similar accuracy for histone modification per round of replication for genomic DNA, no mechanism is conceivable that would achieve a similar accuracy for histone modifications. As a consequence, the amount of stably inheritable epigenetic information is severely limited. Consistent with this theory, most, if not all, of the extraneous epigenetic information is erased during spermatogenesis and oogenesis. The resulting totipotent state (Hackett et al., 2012) is characterized by global erasure of DNA methylation, chromatin reorganization, differential regulation of histone-modifying enzymes (e.g., the tendency for the upregulation of histone de-modifying and downregulation of histone-modifying enzymes). The initial stages of embryogenesis are governed by a gene regulatory network dominated by transcription factors (e.g., reviewed in Adachi and Schöler, 2012), partial ejection of nucleosomes (Watanabe et al., 2013) and therefore a reduction in the availability of a major epigenetic information carrier. Indeed, it seems that only few epigenetic modifications are part of the epigenomic basal state (e.g., strong heterochromatin formation of genes linked to differentiation (Watanabe et al., 2013) or imprinting and poised promoters). In contrast, the error threshold does not preclude inheritance of complex patterns of chromosome marks in somatic cell lines because the number of generations is limited, and usually small. Here, the degradation of the epigenetic information is acceptable for a while, but inevitably leads to daughter cells whose epigenetic patterns are damaged beyond repair. This effect may thus constitute an epigenetic version of aging. Indeed, major changes to the epigenetic information can result in “chromatin diseases” such as cancer (Timp and Feinberg, 2013), for example.

Whether histone modifications and the presence of histone variants are a cause or consequence of the transcriptional status (Henikoff and Shilatifard, 2011; Rando, 2012) is still hotly debated. Here, we are completely agnostic about this issue since it has no impact on our conclusions. The programs that run on the chromatin computer (i.e., the schedules and concentrations of re-writing enzymes) are externally specified in our model. In particular, we make no statement in regards to whether the gene expression program is a direct consequence of, or at least dominated by, the chromatin state, or whether it is entirely determined by classical transcription factor networks that are largely or even completely independent of the chromatin state. In computer science terms, we employ a model of computation that strictly distinguishes between (gene expression) programs and (histone modification) data.

It appears natural, in a next step, to remove this distinction and to ask if chromatin itself can “learn” to re-program itself, by making the gene expression programs an intrinsic function of the histone modification data. While this may be an extreme model that implicitly views transcription factors as being enslaved by histone modification states at their gene loci, it is an important limiting case given that gene expression is clearly not independent of chromatin state.
Acknowledgments

We thank two anonymous reviewers for helpful comments and suggestions. Additionally, we thank Charles L. Nunn and Martin A. Smith for proofreading and Lydia Steiner for stimulating discussions. This work was supported in part by the Templeton Foundation, Grant # 24332 “Origins and Evolution of Regulation in Biological Systems” and the Sächsische Aufbaubank (SAB), Grant # 14494.

References


Watanabe, A., Yamada, Y., Yamakata, S., 2013. Epigenetic regulation in pluripotent stem cells: a key to breaking the epigenetic barrier. Philosophical Transactions of the Royal Society B: Biological Sciences 368 (1605).


