

## Chapter 2

# Optimality Theory and Simulated Annealing

### 2.1 Heuristic optimisation and simulated annealing

#### 2.1.1 Heuristic optimisation for OT

It has been mentioned that Optimality Theory can be seen in two different ways. According to the most frequent picture, the output of GEN is connected to a pipe-line of constraints acting as filters. Alternatively, we may see EVAL as a special function derived from the constraints: the goal is to find the candidate that optimises the Eval-function (also called the *Harmony function*).

Therefore, Optimality Theory belongs to the family of combinatorial optimisation problems. The world is indeed full of optimisation. Entrepreneurs maximise their benefit by minimising costs, football players maximise goals, whereas runners minimise time. Hedonists maximise pleasure, students minimise homework, and so forth. In physics, energy is to be minimised and entropy maximised. In all of these problems, a given function  $f(x)$  has to be optimised (usually minimised, sometimes maximised) subject to a set of conditions, such as  $g_i(x) \geq b_i$  (for  $i = 1, \dots, m$ ) (Reeves, 1995): for what  $x$  satisfying these conditions is the quantity  $f(x)$  optimal?

In Optimality Theory, the *Harmony function*  $H(x)$  should be optimised, subject to the condition  $x \in GEN(UR)$ , and the solution is predicted to be the surface form corresponding to the input underlying representation  $UR$ . Famous members of the family of combinatorial optimisation problems include the *assignment problem* (minimise the costs, if a set of  $n$  people is available to carry out  $n$  tasks, and if person  $i$  performing task  $j$  costs  $c_{ij}$  units); different problems in logistics (planning the routing of vehicles); the *travelling salesman problem* (find a tour of minimum distance that passes through a given set of points, say, towns); or *node colouring* of graphs (find the colouring with the smallest possible number of colours, such that adjacent nodes are not given the same colour).

Certain problems are easy to solve, others are more complex. Sometimes it would require a huge amount of computational resources to find it with cer-

tainty. A great variety of problem-specific or more general solutions have been developed in the recent decades, and this is not the place to give a general overview of them. Here we focus on *heuristic techniques*, and in particular on one of them, namely, *simulated annealing*. Reeves (1995) defines *heuristic* (p. 6) as “a technique which seeks good (i.e. near-optimal) solutions at a reasonable computational cost without being able to guarantee either feasibility or optimality, or even in many cases to state how close to optimality a particular feasible solution is.”

*Simulated annealing* (Reeves, 1995) is one of the simplest and most popular among these heuristic techniques. Simulated annealing, similarly to other heuristic techniques, does not guarantee the correct answer. You do not even know if the answer returned is the optimal one! If you want the probability of obtaining the optimal solution to approach 1, you may require more iterations than exhaustive search (i.e., checking each possibility, supposing a finite search space).

Nevertheless, a heuristic technique such as simulated annealing may have its purpose within linguistics, in general, and within the Optimality Theoretic paradigm, in particular. What are the arguments for finding the optimal element of the candidate set—the element optimising the Harmony function—with simulated annealing, and not with other techniques mentioned in section 1.2?

First, the computation involved in heuristic techniques is simple, not involving a large working or storing capacity—an attractive feature both for cognitive models and for language technology.

Second, you are guaranteed to be returned *some* answer within constant time, even if not necessarily the correct one. In that respect, heuristic algorithms resemble human speech, which also produces outputs at a constant rate, and these outputs are not always fully correct. A counter-argument may be that the brain is an extremely powerful computer and the speech flow is only slowed down by the inertia of the speech organs, this is why we do not observe the time difference between processing simpler and more complex structures. Why then the higher rate of errors for complex structures? Why then the increased number of errors in fast speech, many of which are not due to the inertia of the speech organs? Consequently, I propose to view (some of the) *performance errors*—for instance, in fast speech—as errors introduced by the mental computation. In fast speech, the human mind is willing to give up precision in order to gain speed. And this is exactly a phenomenon that can be reproduced by simulated annealing: not only does simulated annealing return some output certainly within a constant time, but this constant time can be diminished (the simulation can increase its speed) at the cost of precision.

As a matter of fact, I have to acknowledge that sometimes it is hard to distinguish between performance errors that are consequences of computational difficulties and performance errors that are consequences of physical problems. Indeed, we run into the problem of how much phonology is grounded in phonetics: for instance, whether an assimilation process is related to the inertia of some speech organs. So, dropping some segments in fast speech might be argued to result from phonology (which is grounded in phonetics); but it can also be explained within phonetics, even if the phenomenon is influenced by phonological factors. However, progressive and regressive voice assimilations should be equally good solutions from a physical point of view, so if grammar requires regressive assimilation to take place, then a progressive assimilation can definitely

be seen as a result of computational difficulties. Finally, performance errors outside phonology (especially in syntax) cannot be derived from the inertia of the speech organs. In brief, my working hypothesis is that there exist performance errors that are due to computational difficulties.

Additionally, we have seen in section 1.2 that finding the optimal element of the candidate set can be a hard problem: Eisner (2000b) proves that it is NP-hard—worst case exponential—in the size of the grammar. Although many proposals have been advanced to use a series of simple OT-grammars,<sup>1</sup> the traditional view is still to view the language faculty as one huge OT-grammar. In turn, a huge grammar implies computational difficulties that heuristic techniques can overcome the most simply.

### 2.1.2 A technique from statistical physics

Our agenda is, thus, to employ simulated annealing in modelling real-time speech production, including variation or speech errors. In particular, to combine simulated annealing with Optimality Theory. This marriage is, however, far from obvious. In the present subsection, we introduce the key idea of *simulated annealing*, so that we can combine it with Optimality Theory in the next section. The implementations to be presented in chapter 5 will demonstrate the validity of the previous arguments.

*Simulated annealing*, also referred to as *Boltzmann Machines* or *stochastic gradient ascent* (descent), is a wide-spread stochastic technique for combinatorial optimisation, especially in the fields of neural networks (e.g. Reeves (1995), Spall (2003)). The idea originates in solid state physics (Metropolis et al., 1953),<sup>2</sup> and was first presented by Kirkpatrick et al. (1983), as well as, independently, by Černý (1985). It can be also related to the root finding algorithm of Robbins and Monro (1951) (cf. Spall, 2003, p. 97-98).

In linguistics, simulated annealing has been used for (context-free) parsing, by Selman and Hirst (1985), Selman and Hirst (1994) or Howells (1988). Kempen and Vosse (1989) present a cognitive architecture based on activation decay and simulated annealing, and compare the result of simulated annealing with different cooling schedules to aphasic data (see also Vosse and Kempen (2000) for a reconsideration of this model). The link of simulated annealing to connectionist foundations of Harmony Theory—the historical background of OT—goes back as early as Smolensky (1986). However, it has never been used with a concrete linguistic model within OT to my knowledge.

---

<sup>1</sup>*Stratal OT*, a combination of Kiparsky (1982)'s *Lexical Phonology* with Optimality Theory, was already introduced by McCarthy and Prince (1993b). Further examples include, for instance, Bíró and Hamp (2002), who propose a similar model for Israeli Hebrew morphology.

<sup>2</sup>This paper, co-authored by the late Edward Teller, presented a modification of the Monte Carlo integration over the configuration space. The question was how to calculate the average of some quantity  $F$  over a large system. The original Monte Carlo algorithm randomly generated configurations  $x_i$ , in each of which the value of  $F$  was calculated ( $F(x_i)$ ); then, the different  $F(x_i)$ s were averaged with the probabilities  $P(x_i)$  of the configurations, as weights. Nonetheless, the simulation is very likely to generate some of the many improbable configurations, and to avoid the few, high-probability equilibrium states. Therefore, the Metropolis algorithm proposes a random walk in the state space (or phase space), along which the values of  $F$  can be averaged. Indeed, the procedure to be described with respect to simulated annealing produces a series of states  $x_i$  that can be used as a representative sample, that is, in which the frequency of a state  $x_i$  is proportional to its theoretical probability  $P(x_i)$ .

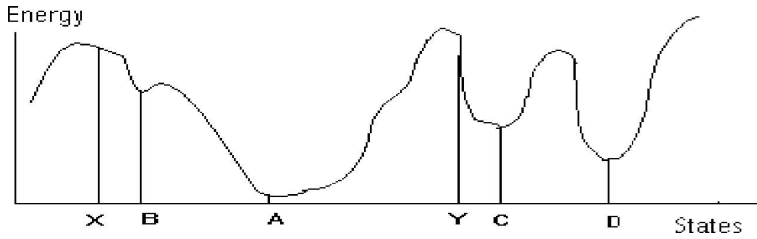


Figure 2.1: Landscape produced by a real-valued energy or cost function (Bíró, 1997).

Simulated annealing is a modification of the algorithm called *gradient descent* (*iterative improvement*; it is also called as *gradient ascent* or as *mountain climbing in the fog*, when used for maximisation). Gradient descent performs a random walk in the search space with the restriction that you may never move upwards, only horizontally, or, preferably, downhill. (For gradient ascent, one simply has to reverse the directions.) At each time step, the random walker picks a neighbour position, compares it with its actual place, and moves there only if the target position is better. Two strategies can be followed: either a neighbour is chosen randomly, or the walker picks its best neighbour, that is, it takes the steepest slope. The position of the random walker at the end of the walk is returned by the algorithm. In both cases, however, the random walker will easily be stuck into local minima—we need therefore a trick to avoid non-global local minima.

Simulated annealing, as we shall see presently, also allows upwards moves, in order to avoid getting stuck into these local minima. The trick originates in statistical physics (thermodynamics and solid state physics).

An interstitial defect in a crystal lattice, say, in the structure of some sort of metal, corresponds to a (non-global) local minimum in the energy of the lattice. Although the perfect lattice would minimise the energy level, the defect is stable, because any local change would increase the energy. In order to climb this energy barrier and to reach the global minimum, one needs either to globally restructure the lattice within one step, or to be permitted to temporarily increase the energy of the lattice. Heating the lattice means to go for the second option. The lattice is allowed “to borrow” some energy, that is, to transform thermic energy provisionally into the binding energy of the lattice, thereby climbing the energy barrier separating the local minimum from the global minimum.

At temperature  $T$ , the probability of a change that increases the lattice’s energy by  $\Delta E$  is  $e^{\frac{-\Delta E}{kT}}$ . Here,  $e \approx 2.7182$  is the base of the natural logarithm, and  $k \approx 1.38 \cdot 10^{-23} JK^{-1}$  is Boltzmann’s constant, which connects energy (measured in *joules*,  $J$ ) to the phenomenological measure of temperature (measured in *kelvins*,  $K$ ). The higher the temperature, the smaller the absolute value of the exponent, and therefore the higher jumps in energy are reasonably probable. At relatively low temperature, the probability of a relatively high jump in energy is not likely to happen.

When annealing a metal, we increase its temperature, therefore the lattice can easily get out of a local minimum. Then, we slowly cool it down. The lower the temperature, the smaller the energy hills the system is able to climb, thus it gets stuck in some valley. Hopefully, this valley corresponds to the global

minimum, but at least to a minimum smaller than the initial local minimum. At the end of the annealing, the system probably arrives at the bottom of a deep valley.<sup>3</sup>

Now, the idea of simulated annealing is straightforward (Kirkpatrick et al., 1983). Suppose that we wish to find the state of a system for which the quantity  $E$  (say, energy or evaluation) is minimal. We define some sort of fictitious “temperature”  $T$ , a mere control parameter, and choose an initial state  $w_0$ , the starting point of a random walk.

At each step of the simulation, when we are in state  $w$ , we randomly choose one of the neighbour states ( $w'$ ) of the actual state. This random choice is one of the main factors driving the stochastic behaviour of the algorithm. It presupposes some sort of *neighbourhood structure (topology)* on the search space that determines which states are the neighbouring states of  $w$ . Moreover, the topology also defines the *a priori* probability  $P_{choice}(w'|w)$  of choosing  $w'$  if we are in state  $w$ . Unless one allows the system not to move at all, we stipulate that

$$\sum_{w' \in \text{Neighbours}(w)} P_{choice}(w'|w) = 1 \quad (2.1)$$

In the simplest case, a state has a finite number of neighbours, and each of them has equal *a priori* probability. We shall come back to further possibilities in section 2.2.2.

Once we have picked a neighbour, we have to decide whether to move there or not. If the neighbour state  $w'$  represents a lower level in  $E$  than  $w$ , we move there. Otherwise, we move only with probability  $e^{-\frac{\Delta E}{T}}$ , where  $\Delta E$  is the increase in energy in the case we took that move:

$$P(w \rightarrow w') = \begin{cases} 1 & \text{if } E(w') \leq E(w) \\ e^{-\frac{E(w') - E(w)}{kT}} & \text{if } E(w') > E(w) \end{cases} \quad (2.2)$$

In other words, a random number with a uniform distribution is chosen from the interval  $(0, 1)$ , and if it is smaller than  $P(w \rightarrow w')$ , then we move from  $w$  to  $w'$ . Why an exponential function? One argument is the analogy from statistical physics, where the probability of state  $s$  at temperature  $T$  is proportional to  $e^{-E(s)/T}$ . Another argument is that the exponential function is the only non-trivial continuous function  $F$  that has the property  $F(x+y) = F(x)F(y)$ ; hence, the probability of moving upwards  $x+y$  is equal to the probability of an uphill step with a difference  $x$  followed by an uphill step with a difference  $y$ .

Please remember the difference between the *a priori* probability  $P_{choice}(w'|w)$  of choosing state  $w'$  when the random walker is in state  $w$ , on the one hand; and the probability  $P(w \rightarrow w')$ , defined in Eq. (2.2), of really moving from state  $w$  to  $w'$ , once  $w'$  has been chosen, on the other. Clearly, the probability of

---

<sup>3</sup>In fact, the energy-structure of traditional lattices is too simple, too trivial, so that the global optimum is usually very well approached. This may be the reason why the original idea, already published by Metropolis et al. (1953), did not raise much interest beyond solid-state physics and related fields for thirty years. The behaviour of spin glasses in magnetic fields, however, yields quite a complicated energy-structure. The investigation of spin glasses led, therefore, both Kirkpatrick et al. (1983) and Černý (1985)—independently of each other—to applying Metropolis’ algorithm on optimisation problems in general, such as the *travelling salesman problem*. For more mathematical details, consult, among others, van Laarhoven (1987).

```

ALGORITHM: Simulated Annealing
Parameters: w_init      # initial state (often randomly chosen)
            t_max      # initial temperature > 0
            alpha       # temperature reduction function

w := w_init ;
t := t_max  ;
Repeat
  Repeat
    Randomly select w' from the set Neighbours(w);
    Delta := E(w') - E(w) ;
    if Delta < 0
      then
        w := w' ;           # move to lower energy state
      else
        generate random r uniformly in range (0,1) ;
        if r < exp(-Delta / t)
          then w := w' ;   # move to higher energy state
        end-if
      end-if
    Until iteration_count = nrep      # usually simply: nrep = 1
    t := alpha(t)
  Until stopping condition = true     # usually: until t < t_min
Return w      # w is the approximation to the optimal solution

```

Figure 2.2: The algorithm of traditional Simulated Annealing.

actually moving from  $w'$  once in  $w$  is the product of the two probabilities. The chance of not moving from  $w$  at all in a certain step is:

$$1 - \sum_{w' \in \text{Neighbours}(w)} P_{\text{choice}}(w'|w) \cdot P(w \rightarrow w') \quad (2.3)$$

Equation (2.2) is the point where the control parameter called “temperature”  $T$  has come into play. Observe that if  $\Delta E = E(w') - E(w) \ll T$ , the probability of moving is practically 1. If, however,  $\Delta E = E(w') - E(w) \gg T$ , the move to  $w'$  becomes almost prohibited. In brief, the role of  $T$  is to define the order of magnitude of  $\Delta E$  in which  $P(w \rightarrow w')$  is neither very close to 1, nor very close to 0.

At the beginning of the simulation,  $T$  is assigned a high value, making any move very likely. The value of  $T$  is then decreased gradually according to some *cooling schedule*, yet  $T$  remains always positive. By the end,  $T$  is very close to zero: according to (2.2), even the smallest jump becomes highly improbable then. When the temperature has reached its lowest value, the algorithm returns the state into which the random walker is finally “frozen”.

The general algorithm of simulated annealing can now be introduced in Fig. 2.2. We follow Reeves (1995), with a few modifications and indications pointing to the way we shall use simulated annealing later.

Why does adding temperature improve the performance of this search algorithm compared to gradient descent? Imagine an asymmetric search space composed of three states, as represented in Fig. 2.3. Suppose that we would

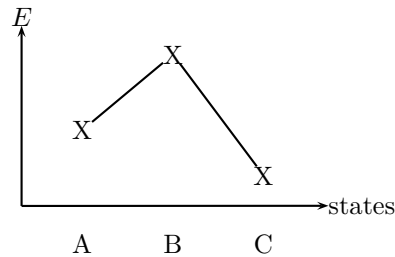


Figure 2.3: An asymmetric landscape with three states, two of which are local optima, but where only state C is a global optimum. State B is a neighbour of both A and C, whereas states A and C are not neighbours of each other.

like to minimise the function  $E$ .

*Gradient descent* (iterative improvement) would assign both local minima 50%. Namely, if each of the states is chosen with equal probability as the initial state, then there is 33% chance that the search begins in state  $A$ ; then, as it is a local minimum, the random walker is stuck there. The same applies to state  $C$ . The remaining one third goes to the case when  $B$  is chosen as the initial state. Then, the two neighbours of  $B$  are chosen with equal probability: whichever is chosen, the random walker moves there, and get stuck there. Summing up, the probability of the search algorithm to terminate in either state  $A$  or state  $C$  is  $1/3 + 1/3 \cdot 1/2 = 1/2$ .

Suppose now that we perform *simulated annealing*. Then, the random walker can climb back from  $A$  or  $C$  to  $B$ . After having climbed to  $B$ , the random walker falls back to  $A$  or  $C$  with equal probability. It is, however, harder to climb to  $B$  from  $C$  than from  $A$  at a given temperature, because  $\Delta E$  is higher.

Roughly speaking, the random walker will be confined to  $C$ , while it still can escape from  $A$ . In order to end up in  $A$ , it has to choose  $A$  always when in  $B$ , otherwise it gets locked into  $C$ . If the number of steps (number of iterations) is  $2n$ , then the random walker can be  $n$  times in state  $B$ : the chance of choosing always  $A$  is  $0.5^n$ , which decreases quickly as  $n$  grows. With higher  $n$ , terminating in  $C$  has a probability  $1 - 0.5^n$  very close to 1. The more iterations, the higher the chance to end up in  $C$ .

Although this train of thought has been only a rough illustration, the general morale still holds: *a slower cooling schedule, that is, increasing the number of the iterations, improves the precision of simulated annealing* (e.g. Reeves (1995), van Laarhoven (1987)).<sup>4</sup> This observation will become very important in the argumentation in favour of Simulated Annealing Optimality Theory.

Obviously, nothing guarantees still that we have found the global minimum. Imagine, indeed, that the global minimum is situated at the bottom of a narrow valley: in such a case, the simulation will most often find a broader, but less deep basin, and the returned state will be the minimum of that one. One can, for example, run a few simulations in parallel, and choose the best of the results of these simulations.

Convergence results exist which prove the chance of finding the global optimum asymptotically converging towards 100% under certain circumstances.

<sup>4</sup>Interestingly, the exact manner of decreasing temperature influences the precision less than the rate at which temperature drops (Reeves, 1995).

Yet, these results imply solution times that are exponential in problem size, and often require more iterations than exhaustive search, so they are of limited use (Reeves, 1995).

### 2.1.3 Spin glasses in the brain

At the end of subsection 2.1.1, we have brought several arguments in favour of using heuristic optimisation techniques in linguistics in general, and for Optimality Theory, in particular. Now, we elaborate on some of them: how can we turn a seeming weakness—the lack of precision—into an advantage. Moreover, how to use it to contribute to a long-standing debate in linguistics, namely, to the issue of competence *vs.* performance, and categorical *vs.* gradient grammaticality.

Kirkpatrick et al. (1983), introduced simulated annealing as an analogy between physical systems and optimisation problems. In particular, they based their hope for the usefulness of simulated annealing upon the success of the Metropolis algorithm (Metropolis et al., 1953) in modelling *spin glasses* in statistical physics (Manrubia et al., 2004, cf. e.g.). Spin glasses are “highly frustrated systems”: magnetic interactions favouring different and incompatible kinds of ordering might be simultaneously present. Determining the optimal state of such a system is far a less trivial optimisation task than finding the optimal structure of a traditional magnetic lattice, and yet the Metropolis algorithm turned out to be useful. Furthermore, Kirkpatrick et al. (1983) write: “*The physical properties of spin glasses at low temperatures provide a possible guide for understanding the possibilities of optimizing complex systems subject to conflicting (frustrating) constraints*” (p. 673). But “conflicting constraints” is exactly the magic key word in Optimality Theory, as well!

A non-negligible difference between spin glasses and OT is, nonetheless, that “*systems like spin glasses have many nearly degenerate random ground states rather than a single ground state with a high degree of symmetry*” (*ibid.*). In other words, while most materials and conventional magnets have only one globally minimal configuration (ground state), which is realised if the particles form a highly symmetric crystal structure; spin glasses, on the other hand, can reach many different local minima, and these minima represent amorphous—glass-like—structures. Reaching a different minimum from some ground state requires considerable rearrangement, hence local minima are quite stable. Last, and most importantly, none of these local optima is significantly worse than the global optimum, thus “*it is not very fruitful to search for the absolute optimum*” (Kirkpatrick et al. (1983), p. 674). As an analogy, simulated annealing is claimed to be the most promising for problems where reaching a near-optimal solution is also satisfactory, such as the physical design of computers (Kirkpatrick et al., 1983) or the travelling salesman problem (Kirkpatrick et al., 1983; van Laarhoven, 1987; Spall, 2003).

In standard Optimality Theory, however, we definitely search for the *global* optimum: the candidate that *globally* optimises Harmony—that is, minimises the violation marks—with respect to the conflicting constraints. In turn, we either hope for a much simpler (less frustrated) search space with an easily reachable global optimum; or we accept the alternative “ground states” as linguistically meaningful solutions. Indeed, we shall see situations for both cases. In section 5, the search space will have a medium complexity: although the

Level	its product	its model	the product in the model
Competence in narrow sense: static knowledge of the language	grammatical form	standard OT grammar	globally optimal candidate
Dynamic language production process	acceptable or attested forms	SA-OT algorithm	local optima
Performance in its outmost sense	acoustic signal, etc.	(phonetics, pragmatics)	??

Table 2.1: The proposed three-level model of the human language

global optimum is easy to find with a slow cooling schedule, yet a fast cooling schedule may return other local optima, which are interpreted as fast speech forms. In section 6.1, however, some local optima will be interpreted as well attested (and acceptable) agrammatical forms. What is meant here?

Remember the Chomskyan distinction between *acceptability* and *grammaticality* (section 1.4): grammaticality refers to the linguistic competence or to its model, and not to the performance. As far as performance is concerned, a native speaker usually distinguishes between several levels of acceptability, even if some of the forms in the grey area are clearly grammatical or clearly agrammatical. Similarly, a corpus study will reveal that agrammatical forms are well attested, whereas grammatical ones may be very rare. This is why I urged in section 1.4 a model that makes the difference between the underlying static knowledge of the language (competence in its narrow sense), the dynamic production process (which may produce—or accept—forms that are agrammatical for the competence in its narrow sense) and the clearly extra-linguistic factors.

This is where the numerous near-optimal ground states of the spin glasses can be used as an analogy. I propose to see *Optimality Theory*, the model that formulates the optimisation problem, as the model of competence in its narrow sense. A form is *grammatical* if and only if it is the optimal solution to the problem posed by the OT grammar—that is, if it is *globally* optimal. On the other hand, further local optima also appear in the model, which will serve as pitfalls to *Simulated Annealing for Optimality Theory*, the model of the dynamic language production process. Consequently, the linguistic interpretation of the non-optimal ground states is that they model linguistic forms that are *agrammatical but acceptable* according to the judgement of the native speaker; alternatively, they are the forms that are *agrammatical but attested* in a corpus study (Table 2.1).

This approach—I believe—has two advantages. First, it may save a never-ending discussion on where the exact border between competence and performance is to be drawn. Second, it may help keep the competence model simple, as to be demonstrated by section 6: only the general rule has to be accounted for by the grammar (the model of the competence in the narrow sense), and (some) exceptions can be explained as being introduced by the language production process. Remember the history of mechanics: the decomposition of the problems into friction and motion following Newton’s laws helped grasping the sublunar motions, as well.

Now the question is how to implement Simulated Annealing to Optimality

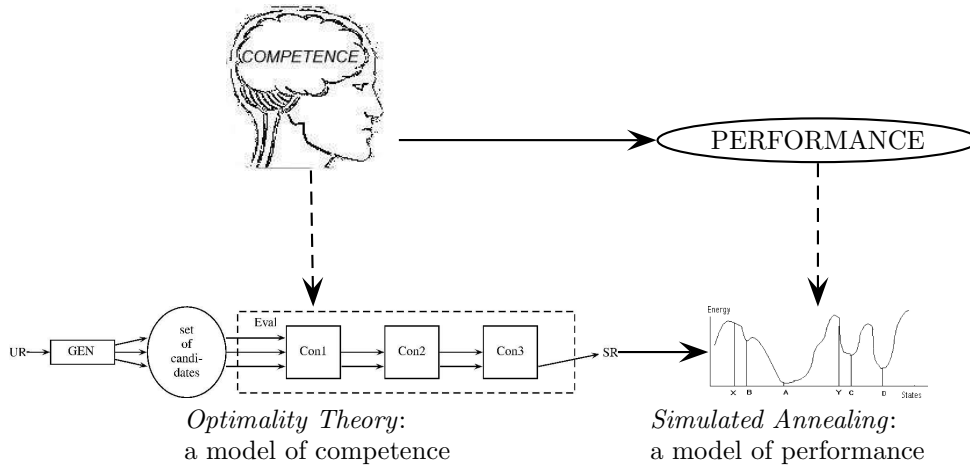


Figure 2.4: A simplified picture of the proposed relationship of OT and SA-OT.

Theory exactly. The search space is the set of candidates, but a real-valued energy function to be minimised cannot be defined in most cases (Prince and Smolensky (2004); see Prince (2002) for models where it can). Therefore, we have to find out how to implement simulated annealing to a non-real valued function. Section 2.2 introduces the general idea first. (Later, Chapter 3 will present a much more formal way to the same algorithm, by asking the question: if the function to be optimised is not real-valued, what is it then?) The algorithm of simulated annealing presented in Fig. 2.2 will have to be adjusted to the answer. After having constructed the SA-OT algorithm, section 2.3 contains a discussion of cases where SA-OT works and where it does not work as we might expect in anticipation.

## 2.2 Simulated Annealing for Optimality Theory

### 2.2.1 How to combine simulated annealing with OT?

Let us summarise where we are so far. Generating the winner candidate in Optimality Theory, as even suggested by its name, is a combinatorial optimisation problem. The goal is *to find the optimal candidate* in the candidate set, with respect to some *Harmony function* defined by the constraint hierarchy. It can be an NP-hard problem (Eisner, 1997, 2000b; Wareham, 1999), which motivates the use of heuristic algorithms in general, and the use of simulated annealing in particular.

The general idea of *Simulated Annealing for Optimality Theory* (SA-OT) is that the random walker roves around in the search space, which is the candidate set. The possible states of the system are thus the candidates. The function to be optimised is the Harmony function. Yet, the algorithm presented in Fig. 2.2 requires a few more non-trivial details so that we may implement it. What is  $Neighbour(w)$ ? How to calculate the difference  $E(w') - E(w)$ ? What should the cooling schedule ( $t_{max}$ , the stopping condition or  $t_{min}$ , and the function

$\alpha(t)$  be in SA-OT?

The procedure of applying simulated annealing to Optimality Theory will be decomposed into the following five steps:

- Step 1: Define the candidate set.
- Step 2: Define a neighbourhood structure (topology) on the candidate set.
- Step 3: Define the Harmony function to be optimised: what are the constraints and how are they ranked?
- Step 4: Define temperature and the transition probabilities.
- Step 5: Define the cooling schedule and perform the simulation.

Steps 1 and 3 are familiar to anybody who has worked with Optimality Theory. Yet, the formal definition of the candidate set and of the constraints may require some extra work. In most of the linguistic literature, Gen is vaguely seen as a “black box” that produces “everything”. For computational purposes, nonetheless, Gen (that is, the candidate set) has to be defined in an exact way: for instance, by explicitly listing the elements of the set, by specifying clear conditions on membership,<sup>5</sup> or by using finite-state automata or context-free grammars. Moreover, constraints are usually defined by specifying the conditions that must be met by a candidate to *satisfy* the constraint. Yet, OT constraints are violable, and often the degree of violation plays a crucial role. Therefore, computational implementations require the formulation of constraints as functions that specify clearly how many violation marks are assigned to each candidate. For instance, a frequently employed, though only vaguely defined constraint—which we will reformulate for the sake of SA-OT—is Output-Output Correspondence: while the OT phonological literature is pleased by qualitatively demonstrating that a certain candidate is worse for OOC than the optimal one, SA-OT will require the exact difference of the marks assigned to any pair of candidates.

The hierarchy of the constraints should be settled by linguists working within the traditional OT paradigm, because the meaning of the hierarchy remains unchanged: to return the candidate that is supposed to be the grammatical one. The model underlying SA-OT is a traditional Optimality Theoretic grammar. It remains the task of the linguist to collect the data (that is, to find out which forms are grammatical), to make cross-linguistic comparisons, and to argue for specific constraints and specific hierarchies.

In brief, Gen, the set of constraints and the ranking—handed over by the theoretical linguist to the computational linguist—should yield as the optimal candidate the grammatical form, which is observed empirically by the descriptive linguist. What remains to clarify from the above agenda is step 2, that is, defining the neighbourhood structure (the topology); as well as, steps 4 and 5, the actual implementation of simulated annealing.

---

<sup>5</sup>As it is the case in all fields of mathematics, such a definition of a set must have the following form: “all members of set  $S$  [another well-defined set] that satisfy these well-defined conditions”. The goal of the exact definition is to be able to handle them in an algorithm.

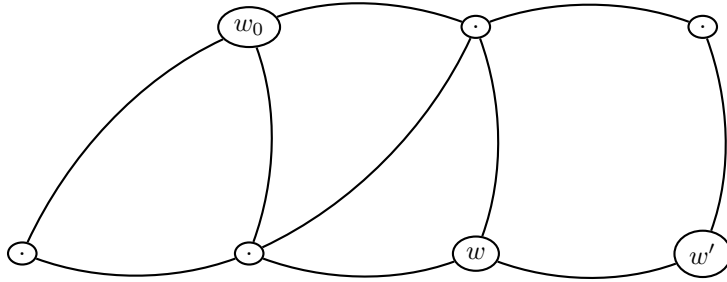


Figure 2.5: Schematic view of a search space—that is, the candidate set with a simple topology (neighbourhood structure)—in which simulated annealing realises a random walk. Candidate  $w_0$  is the initial state of the random walk. Candidate  $w'$  is a neighbour of  $w$ , as they are connected by an edge.

### 2.2.2 Topology on the search space

We shall come back to step 4 in subsection 2.2.3. Our goal now is to work out the *neighbourhood structure* (the *topology*) of the candidate set (step 2), so that performing the simulation with different cooling schedules (step 5)—or with different parameter settings, in general—return interesting results.

The word *topology* refers to “the mathematical study of the properties that are preserved through deformations, twisting, and stretching of objects. Tearing, however, is not allowed.”<sup>6</sup> The classical such property is *neighbourhood*: neighbouring points remain neighbours during twisting and stretching (but not during tearing), even though their distance may change. Thus, distance in absolute terms does not interest topology. A forerunner of topology was *graph theory*, which is exclusively interested in the connection between the vertices, but not in their spacial positions: moving the nodes of a graph does not alter it, as long as the edges are kept the same. If one prefers, one may employ the term *geometry of the candidate set*, as well.

When speaking about the *topology* or the *neighbourhood structure* of the candidate set (the search space), we have to imagine a graph-like structure in a first approximation (Fig. 2.5). Although the candidate set can include an infinite number of elements, its structure may be visualised as a set of points with the neighbouring points being connected by edges. In addition, the edges on this graph-like picture may be directed and labelled in order to represent the probabilities of picking a given neighbour.

In section 2.1.2, we have already seen what the topology on a search space consists of. For each state (now, candidate)  $w$ , we define the set  $Neighbours(w)$  of its neighbours. A directed edge can be drawn from vertex  $w$  to vertex  $w'$  if and only if  $w' \in Neighbours(w)$ . Supposing that the neighbourhood relation is symmetric, one does not even need a directed graph. Importantly, this “graph” must be connected: a path of a finite length should connect any two vertices.

Furthermore, we have also seen that we also require a probability distribution on the set  $Neighbours(w)$ : the *a priori* probabilities  $P_{choice}(w'|w)$  determine the choice of a particular neighbour in the first line of the core of the loop in the

<sup>6</sup>Eric W. Weisstein. “Topology.” From MathWorld—A Wolfram Web Resource. <http://mathworld.wolfram.com/Topology.html>.

simulated annealing algorithm (Fig. 2.2). These probabilities can be written on the directed<sup>7</sup> edges of the graph—or graph-like structure, if the number of candidates is infinite. Due to equation (2.1), the sum of the weights leaving each vertex is 1.

Notice that the topology of the candidate space is the *horizontal structure* of the landscape in which the random walk takes place, and is independent of the landscape’s *vertical structure*, defined by the Harmony function. That is to say that the same candidate set with a different hierarchy will yield the same  $P_{choice}(w'|w)$  *a priori* probabilities, and different  $P(w \rightarrow w'|T)$  transition probabilities. The former is defined by the candidate set, whereas the latter depends on the violation profile (that is, the “altitude”) of the two candidates.

It should be emphasised that adding a structure to the candidate set is new within Optimality Theory as seen by (almost) all linguists.<sup>8</sup> I postulate that this structure is universal (innate), the same way as the set of possible underlying representations (*Richness of the Base*, Prince and Smolensky (2004) p. 225) and the Gen module—thus, the set of candidates—are claimed to be universal (innate). I suppose that the topology of the search space is a result of the way the candidates are represented, and neighbours differ only in a minimal component of their representations.<sup>9</sup>

Although the structure of the candidate set is assumed to be universally defined (if you wish, innate), it is still unknown to us. Research should discover it. That is to say, models have to be created to describe empirical data. Well, one may object that enriching the OT model by adding new components (such as the topology) to it will lead to *ad hoc* models that are less convincing. Indeed, there is such a danger: Ockham’s razor advises simplifying scientific models, and not adding new concepts to them.<sup>10</sup> Nonetheless, topology is not a superfluous addition, and two factors keep a tight rein on it. First, it should be convincing and cannot be *ad hoc*: a general principle has to define what simple basic operations (possible basic steps in the candidate set) transform one candidate into its neighbour. Second, the enriched model is required to account for an enriched set of data: not only for what is grammatical, but also for what the alternations or performance errors are, and under what circumstances they appear in speech.

---

<sup>7</sup>If the neighbourhood relation is symmetric, the graph is not directed in the sense that  $w_1$  is connected to  $w_2$  if and only if  $w_2$  is connected to  $w_1$ . However,  $P_{choice}(w_2|w_1) = P_{choice}(w_1|w_2)$  does not necessarily hold even in this case.

<sup>8</sup>Paul Smolensky mentioned in a talk (October 2004, in Amsterdam) that variation forms are *local optima*—exactly what the present research line is about. Therefore, he must also suppose some topology on the candidate set, otherwise the expression *local optima* would be meaningless. Furthermore, the candidate set has been supposed to have the structure of a regular grammar in finite state approaches to OT since Ellison (1994).

<sup>9</sup>This proposal bears clear similarity with Paul Smolensky’s connectionist approach to Optimality Theory. Yet, here we are free to embrace also representations different from those advanced by connectionism.

<sup>10</sup>*Ockham* was a nominalist in the medieval dispute between nominalists and realists. Realists (today, we would call them idealists) supposed that the Platonian ideals existed ontologically, whereas nominalists (e.g. Abelard) claimed that concepts are created only by the human intellect. According to Ockham, “plurality is not to be posited without necessity”, and referred to the razors used in those days to remove unnecessary ink from pergaments. Thus, nowadays, we should rather refer to *Ockham’s eraser* or to *Ockham’s delete button*, as proposed by Gábor Balázs—to whom I thank for this explanation. See also: Spade, V.S., *Ockham’s nominalist Metaphysics*, In: Spade, V.S. (ed.), *The Cambridge Companion to Ockham*, 1999. pp. 101-102.

Simulated annealing makes a mistake when it gets stuck in a *local optimum*—supposing that the simulation has enough time to relax in its final phase. (Otherwise, if the simulation is terminated without waiting for it to arrive into some local optimum, basically any form may be returned.) Consequently, the topology should be defined so that the observed alternation forms be local optima: even if they are not globally optimal, they must be better than all their neighbours. The horizontal component (neighbourhood structure) and the vertical structure (Harmony function) of the landscape should jointly cause the simulation to sometimes fall into these traps.

The art of *Simulated Annealing Optimality Theory*, thus, consists of creating a landscape—with a simple and convincing definition of the neighbourhood structure and with arguably universal constraints—where the global optimum is the grammatical form and the other local optima are the observed alternations.

Later sections will introduce models demonstrating that even having such a landscape is not always sufficient. For instance, some local optimal traps are always avoided, or are never avoided. Sometimes, the logic of the representations forces us to also include other local optima: in such cases, we can only hope that they are avoided by the random walker, while the local optima corresponding to observed alternation forms are those where the walker is sometimes caught. By the end of the present thesis, the reader should be convinced that adding a topology to the candidate set does increase the explanatory power of Optimality Theory, because it may account for observations in a non-trivial way.

Now, let us elaborate on the concept of a topology on the candidate set. The goal of the neighbourhood structure, again, is to define how a next candidate  $w'$  is chosen as a *possible* next state of the random walker, when the walker is in candidate  $w$ . Obviously, after having chosen  $w'$ , it is still not sure yet that the walker really moves there: this depends on the transition probability  $P(w \rightarrow w'|T)$ , to be defined in the next section following equation (2.2).

As a matter of fact, we should distinguish between three steps when introducing the topology of a candidate set:

1. Define the set of candidates
2. Define the set of the neighbours  $Neighbours(w)$  of each candidate  $w$ .
3. Define the *a priori* probability distribution  $P_{choice}(w'|w)$  on  $Neighbours(w)$ .

What we really need is the probability distribution  $P_{choice}(w'|w)$ , yet the first two steps will lead to it. As it reflects some sort of probabilistic connection between states, one can compare this probability distribution to a Markov-model with two major differences: the number of states may be infinite, and choosing  $w'$  does not mean immediately moving there. Rather the product  $P_{choice}(w'|w) \cdot P(w \rightarrow w'|T)$  is the probability of really moving from  $w$  to  $w'$ ; but then, this probability changes during the simulation as  $T$  changes, hence, we cannot speak of a Markov chain, either.

All introductions to simulated annealing emphasise the importance of an adequate neighbourhood structure in order to obtain an efficient optimisation algorithm. Too few neighbours, on the one hand, may result in too many local optima, and finding the global optimum becomes improbable. Too many neighbours, on the other, turns the algorithm practically into a random or exhaustive search, and does not deploy the structure of the search space.

A few strategies can be followed, and we turn now to comparing them. What has been suggested until now is that you should best define some *basic operations* which transform a candidate to a very similar other candidate. These operations typically alter only the candidate string at one single point, for instance by inserting, deleting or rewriting one atomic element of the string, or by flipping the value of a single feature. The number of *basic operations* should be minimal and the definition of such a *basic operation* (*basic step*, *basic transformation*) has to fit the way candidates are represented in the grammar (autosegmental phonology, context-free trees, AVMs, and so forth). Such *basic steps* would then lead to only a very restricted number of neighbours:<sup>11</sup>

$$\text{Neighbours}(w) = \left\{ w' \in \text{GEN}(\text{GEN}^{-1}(w)) \mid w' = \text{some\_basic\_transfo}(w) \right\} \quad (2.4)$$

Nevertheless, even if the strategy allows only one basic operation per step, more options are available with respect to probabilities. One could assign *ad hoc* probabilities to the neighbours, but two further alternatives are more sound.

Either each of the neighbours has equal chance to be picked out—this proposal sounds reasonable if every candidate only has a few neighbours. We shall apply this approach in our treatment of Dutch stress in fast speech in Chapter 5, as well as in our example about Dutch voice assimilation in Chapter 6.1. If  $\#$  denotes the cardinality of a set,

$$P_{\text{choice}}(w'|w) = \begin{cases} \frac{1}{\#\text{Neighbours}(w)} & \text{if } w' \in \text{Neighbours}(w) \\ 0 & \text{else} \end{cases} \quad (2.5)$$

Alternatively, performing each of the basic steps can be assigned some probability, and thus not all neighbours have equal chance. For instance, we first decide whether to insert, to delete or to rewrite an atomic segment, and then decide where in the string to perform this action. Even if each locus in the candidate string has equal chance, the probabilities  $p_{\text{insert}}$ ,  $p_{\text{delete}}$  and  $p_{\text{rewrite}}$  may vary. An example for this approach is found in our discussion of syllabification in section 7.

So far,  $\text{Neighbours}(w)$  has been a relatively small subset of the candidate set. As already mentioned, the emerging “graph” must be a connected structure: each candidate should be reachable from any other candidate within a finite number of steps. In other words, for all  $w$  and  $w'$ , there must exist an integer  $n$  and a series of candidates  $w_0, w_1, \dots, w_n$  such that  $w_0 = w$ ,  $w_n = w'$ , and for each  $j < n$ :  $w_{j+1} \in \text{Neighbours}(w_j)$ .

<sup>11</sup>The implementation of Optimality Theory by Turkel (1994) uses a genetic algorithm, and proposes to see OT Gen as the generator of a new GA generation. He writes (p. 8):

[t]he standard assumption about the generator is that it takes a single representation and returns a set of representations consisting of modifications to the input. I will assume that the generator takes a set of representations and returns a set of representations. If the input set contains one element, then the generator returns a number of variations on that element (this is the standard operation). If the input set is empty, then the generator randomly creates a set of appropriate representations and returns that. ...

His standard operations (modifications, such as mutations, recombinations and crossovers) correspond to our *basic operations*: by applying them repeatedly, we can explore the search space. These operations, as proposed by several readers, could also be underpinned psycholinguistically. In any case, future work has to work out some general principles.

A radically different approach is to define the set of neighbours as the whole candidate set:

$$\text{Neighbours}(w) = \text{GEN}(\text{GEN}^{-1}(w)) \quad (2.6)$$

and to focus rather on the probabilities. In this case, some sort of similarity measurement may serve as  $P_{\text{choice}}(w'|w)$ . The more similar  $w$  and  $w'$ , the higher the probability  $P_{\text{choice}}(w'|w)$ . The similarity measurement must be normalised, so that  $\sum_{w' \in \text{GEN}(\text{UR})} P_{\text{choice}}(w'|w) = 1$  hold (cf. equation (2.1)).

An approach based on (2.6) allows huge steps, which may be both useful and harmful. Clearly, if each candidate is equally reachable from a certain candidate, then simulated annealing turns into a very clumsy and ineffective way of trying out all possibilities: many candidates will be tried out repeatedly, whereas many other candidates will be ignored (let alone what happens in an infinite search space). Consequently, the  $P_{\text{choice}}(w'|w)$  probabilities should make use of the properties of the search space in order to direct the search in a clever way. Indeed, a similarity-like  $P_{\text{choice}}(w'|w)$  sounds promising, although further experimentations will be required.

In fact, a topology where every two candidates are neighbours, has no local optima, except for the global optimum. But notice that for all  $\epsilon > 0$ , the set  $\{w' | P_{\text{choice}}(w'|w) > \epsilon\}$  defines a finite  $\epsilon$ -neighbourhood around  $w$ .<sup>12</sup> This observation becomes important when applying a neighbourhood (2.6) to an infinite (or very large) set. Namely, if  $1/\epsilon$  is in the magnitude of, or greater than the number of iterations performed, then the candidates beyond the  $\epsilon$ -neighbourhood of  $w$  are practically unreachable from  $w$  within one step. In brief, such an infinite neighbourhood can be elegant, and still not significantly different from the finite neighbourhood structure defined by equation (2.4).

What do I mean by an elegant model? Sometimes, allowing one single basic operation per step, as in equation (2.4), may lead to problems. The example on syllabification in Chapter 7 will show that allowing exclusively the simplest basic steps may not prove very useful: the landscape will include too many local optima. There, one also has to be able to delete fully overparsed syllables—and not only single segments—in order to build a workable model. In similar cases, it is more fruitful to include *ad hoc* larger steps, as well. In turn, a well-designed model in which  $P_{\text{choice}}(w'|w) > 0$  for any pair of candidates might prove to be both more elegant and useful.

Earlier in this section, I postulated the topology of the candidate set to be universal (alternatively, innate). Now, I should add that although the principles of the topology are claimed to be universal, yet some parameters may be language-dependent or speaker-dependent. Take the approach in which different basic transformations might have different probabilities: for instance,  $p_{\text{insert}}$ ,  $p_{\text{delete}}$  and  $p_{\text{rewrite}}$ , which are not dependent on the argument of the operation. These three parameters are not independent of each other,  $p_{\text{insert}} + p_{\text{delete}} + p_{\text{rewrite}} = 1$  should hold, because exactly one of them can be applied in one step. Now, the fact that three basic operations with some probabilities as parameters define the topology is claimed to be universal; nonetheless, the exact value of these parameters may vary across speakers or across languages.

<sup>12</sup>As  $\sum_{w' \in \text{GEN}(\text{UR})} P_{\text{choice}}(w'|w) = 1$  must hold, fewer than  $1/\epsilon$  candidates may be within such an  $\epsilon$ -neighbourhood.

In this section, we have elaborated on the concept of a neighbourhood structure on the candidate set. The concept is new for main-stream Optimality Theory, even if not for connectionist approaches to OT. We have proposed several possibilities to define this topology, but only concrete applications will prove the usefulness of any of them. In the following section, we tackle the problem of how to define *temperature* for Simulated Annealing Optimality Theory.

### 2.2.3 Temperature for OT

As explained on page 45, creating an SA OT model consists of the following steps:

- Step 1: Define the candidate set.
- Step 2: Define a neighbourhood structure (topology) on the candidate set.
- Step 3: Define the Harmony function to be optimised: what are the constraints and how are they ranked?
- Step 4: Define temperature and the transition probabilities.
- Step 5: Define the cooling schedule and perform the simulation.

Out of these five steps, step 1 and step 3 depend on the traditional Optimality Theoretic system that serves as the underlying model. Subsection 2.2.2 has elaborated on step 2. Our present task is to work out step 4, so that we can experiment with different models (a candidate set with a neighbourhood structure, as well as a set of constraints with a hierarchy) and different cooling schedules.

The goal of defining a temperature is to define the transition probabilities  $P(w \rightarrow w'|T)$  of moving in a counter-optimal direction. Remember that stepping to a more optimal candidate should always be possible: once a certain neighbour  $w'$  of the present position  $w$  of the random walker has been chosen, the random walker moves there if  $w'$  is better than  $w$ .

If  $w' \in \text{Neighbours}(w)$  and  $w' \succ w$ , then  $P(w \rightarrow w'|T) = 1$  for all  $T$ .

Here, and from now on, the relation  $a \succ b$  denotes that candidate  $a$  is *better than* candidate  $b$  with respect to the current constraint hierarchy. A more formal definition follows in section 3.1.

In brief, the  $w' \succ w$  case is simple. Yet, if  $w' \prec w$ , the probability of moving to  $w'$  should depend on the loss of harmony that this move would involve: the steeper the step, the less probable it is. Moreover, the probability of a particular step is gradually diminished from 1 to 0 during the simulation. The parameter called “temperature” is introduced to simulated annealing exactly in order to control the way how each of the probabilities  $P(w \rightarrow w')$  are gradually reduced from 1 to 0.

Let us recapitulate the idea of simulated annealing. The notion of temperature is taken from thermodynamics and statistical physics. A physical system at a temperature above absolute zero has some inner random “vibration”, and this thermic energy “flows” between the particles of the systems as they interact with each other. Consequently, the energy of a given particle may randomly increase and decrease. Emitting an energy quantum is always possible (hence,

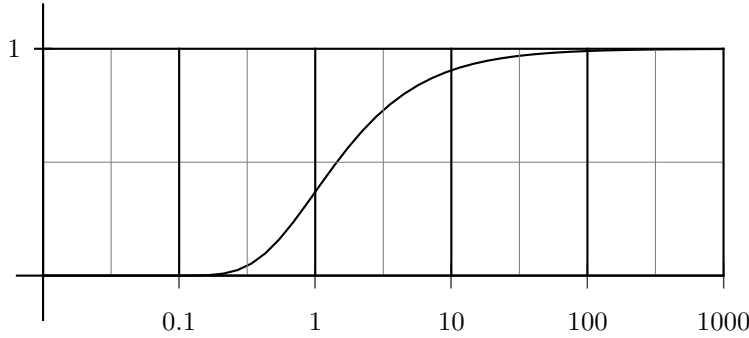


Figure 2.6: The  $e^{-1/x}$  function with a logarithmic  $x$ -axis. Observe that the function is very close to 0 if  $x < 0.1$ , and very close to 1, if  $x > 100$ .

moving towards the better state has a probability of 1), whereas the chance of collecting, or borrowing, an energy package of  $\Delta E$  depends on the temperature  $T$  of the system:

$$e^{-\frac{\Delta E}{k \cdot T}} \quad (2.7)$$

where  $k$  is *Boltzmann's constant* ( $k = 1.3807 \times 10^{-23} J \cdot K^{-1}$ ;  $J$  stands for *joules* and  $K$  for *kelvin*), whose role is merely to connect energy to temperature, so that the exponent has no unit of measurement.

Let us have a closer look at the expression (2.7), as well as at Fig. 2.6. The meaning of temperature  $T$  is to define the *range* of energy transitions that have an intermediate probability. Large jumps in energy ( $\Delta E \gg kT$ ) have a vanishingly small probability, very close to zero; whereas small changes in energy ( $\Delta E \ll kT$ ) are almost certain to happen. For  $\Delta E = kT$ , however, the probability is  $1/e \approx 0.37$ . These observations are summarised in Table 2.2.

Therefore, defining temperature in simulated annealing means defining the transitions that we want to assign this medium probability to in a certain stage of the simulation. Thus, temperature has to belong to the same “type” as the changes in the function to be optimised. In physical terms, the exponent has to be dimensionless, that is,  $\Delta E$  and  $k \cdot T$  in (2.7) must have the same units of measurement (for example *Joule* or *kcal*).

This is also the reason for the introduction of Boltzmann's constant in physics. For historical and human reasons, the temperature scale has been defined independently of energy: we perceive temperature as a *qualia* in its own right, even though physics has reduced this concept to the concept of energy. In fact, why not call rather  $k \cdot T$  the temperature? This is the way simulated annealing proceeds. In simulated annealing,  $k = 1$  simply, so energy and temperature have the same dimension.

Our goal is to implement equation (2.2) for Optimality Theory in a form such as:

$$P(w \rightarrow w' | T) = \begin{cases} 1 & \text{if } w' \succ w \\ e^{-\frac{H(w') - H(w)}{T}} & \text{if } w' \prec w \end{cases} \quad (2.8)$$

$\Delta E < 0$	$P(w \rightarrow w' T) = 1$
$0 < \Delta E \ll T$	$P(w \rightarrow w' T) \approx 1$
$\Delta E \approx T$	medium $P(w \rightarrow w' T)$
$\Delta E = T$	$P(w \rightarrow w' T) = 1/e \approx 0.368$
$\Delta E \gg T$	$P(w \rightarrow w' T) \approx 0$

Table 2.2: The way the transition probability  $P(w \rightarrow w'|T)$  is dependent on the temperature  $T$  and on the steepness  $\Delta E = E(w') - E(w)$  of the transition  $w \rightarrow w'$ . By decreasing the positive control parameter  $T$  during the simulation, the same transition turns gradually from being highly probable into being highly improbable.

Consequently, we need to know the dimension of the Harmony function, in order to introduce temperature  $T$  to SA-OT. So, what is the *dimension* of the Harmony function?

The situation is even worse: not only does the Harmony function lack a proper dimension, but normally it is even not construed as a real-valued function! How can we divide the Harmony function with anything, and then compute its exponential?

Luckily enough, however, we do not need the dimension of the Harmony function proper. What we really need in equation (2.8) is the *difference*  $H(w') - H(w)$  of two violation profiles. This seemingly additional step—subtraction—will help us. For this reason, our action plan to define  $P(w \rightarrow w'|T)$  for the case  $w'$  is worse than  $w$  will be the following:

- Firstly, we represent the violation profiles in an appropriate way.
- Secondly, we define the *difference* of two violation profiles.
- Thirdly, we define *temperature* in a similar format.
- Fourthly, we define the exponential of their quotient.
- Lastly, we introduce the SA-OT algorithm.

We shall perform this action plan several times. In this section, we try to build up an intuitive idea. Therefore, we shall represent violation profiles as vectors, whereas the difference of two violation profiles will be a pair of numbers. A violation profile as a vector is but a shorthand for a row in a traditional tableau, most probably familiar to the reader. Subsequently, the following section presents two alternative mathematical models. Yet, all three approaches will lead to the same SA-OT algorithm.

### Difference of violation profiles

We follow the action plan just presented, and begin with the representation of a violation profile. In traditional terms, a violation profile is a set of tokens of violation marks. For instance, candidate  $w_1$  incurs two marks from constraint C4, one mark from C2, five marks from C1, and one mark from constraint C0.

Such a set can be simply visualised by a tableau, many of which we already saw in Chapter 1.

If candidate  $w_1$  has incurred five violation marks from constraint C1, then we simply write  $C_1(w_1) = 5$ : candidate  $w_1$  has a *violation level* of 5 on constraint C1. In short, constraints are functions mapping from the set of candidates onto the set of non-negative integers.

Suppose that the constraints form the following hierarchy:  $C_N \gg C_{N-1} \gg \dots \gg C_0$ . Then, a row in a tableau is:

$$\begin{array}{|c|c|c|c|c|} \hline & C_N & C_{N-1} & \dots & C_0 \\ \hline w & C_N(w) & C_{N-1}(w) & \dots & C_0(w) \\ \hline \end{array} \quad (2.9)$$

Frequently, cells with value  $C_i(w) = 0$  are left empty. Now, a violation profile can be seen as a vector formed by the levels of violation:

$$H(w) = (C_N(w), C_{N-1}(w), \dots, C_0(w)) \quad (2.10)$$

It is simply a shorter way to write a row of a tableau. We shall call the  $H(w)$  thus introduced *the violation vector* corresponding to the *violation profile* of  $w$ . It will be also called the *vector representation* of the Harmony function.

Note that the indices *decrease* within the vector representation. The first component of the vector has the highest index, and it represents the violation level of the highest ranked constraint. Although this notation might look awkward at this point, it will become handy later on to assign higher indices to higher ranked constraints. Additionally, this notation clearly parallels an OT tableaux, in which the highest ranked constraints appear on the left.

Now, we proceed to the *difference* of two violation profiles. Let us take two violation profiles represented in the form of a traditional tableau:

$$\begin{array}{|c|c|c|c|c|c|} \hline & C4 & C3 & C2 & C1 & C0 \\ \hline w_1 & ** & & * & ***** & * \\ \hline w_2 & ** & & *** & & * \\ \hline \end{array} \quad (2.11)$$

The reader familiar with Optimality Theory will immediately observe that the crucial constraint that *differentiates* between the behaviour of the two candidates  $w_1$  and  $w_2$  is constraint C2. We shall call a constraint playing this important role the *fatal constraint*, the *critical constraint*, or, following Prince and Smolensky (2004), *the highest ranked constraint with uncanceled marks*.

Even though candidates  $w_1$  and  $w_2$  do not necessarily satisfy the two highest ranked constraints, C4 and C3, these constraints do not play any role in the comparison. If the question is whether to move from  $w_1$  to its neighbour  $w_2$ , the two violation marks incurred by both candidates with respect to constraint C4 just “elevate the baseline”: hiking from 500 m above sea-level to 800 m above sea-level is the same as hiking from 1500 m to 1800 m above sea-level. In brief, these highly ranked, but shared violations will not influence the difference in the Harmony value of the two candidates.

Furthermore, standard Optimality Theory teaches us not to look further in the hierarchy once we have found a difference between the violation profiles (for counter-examples and the notion of *cumulativity*, see subsection 1.3.5). Candidate  $w_2$  is defeated by  $w_1$  at constraint C2, and the many violation marks assigned to  $w_1$  by C1 do not make any *difference*. Hence, the *difference* of these

two violation profiles will be “two violations of constraint C2”. That is, the difference of two violation profiles will have the form of a pair: a constraint followed by the difference of the violation levels of this constraint.

By generalising this concrete example, we now define the *difference of two violation profiles*, by following the theoretical foundations of OT. Prince and Smolensky (2004) introduce the concept of *mark cancellation* and prove the *Cancellation Lemma* (p. 258):

**Cancellation Lemma.** Suppose two structures [*violation profiles*—*T.B.*]  $S_1$  and  $S_2$  both incur the same [*violation*] mark \*m.<sup>13</sup> Then to determine whether  $S_1 \succ S_2$ , we can omit \*m from the list of marks of both  $S_1$  and  $S_2$  (‘cancel the common mark’) and compare  $S_1$  and  $S_2$  solely on the basis of the remaining marks. Applied iteratively, this means we can cancel *all* common marks and assess  $S_1$  and  $S_2$  by comparing only their unshared marks.

A consequence of this lemma is the *Cancellation/Domination Lemma* (Prince and Smolensky (2004) p. 261):

**Cancellation/Domination Lemma.** Suppose two parses [*candidates*]  $A$  and  $B$  do not incur identical sets of marks. Then  $A \succ B$  iff every mark incurred by  $A$  which is not cancelled by a mark of  $B$  is dominated by an uncancelled mark of  $B$ .

It is this second lemma that teaches us that the crucial point in comparing two candidates is the highest constraint  $C_{fatal}$  where the two profiles differ. All violation marks assigned by constraints higher than  $C_{fatal}$  are cancelled, whereas lower violation marks are dominated by some violation of  $C_{fatal}$ . Consequently, constraints ranked lower than  $C_{fatal}$  can be ignored.

For the sake of simulated annealing, however, we require not only the *comparison* of two violation profiles, but also their *difference*. The general “philosophy” of Optimality Theory just presented motivates us to neglect what happens at constraints lower than the fatal constraint:

**Definition 2.2.1.** DIFFERENCE OF TWO VIOLATION PROFILES: *Suppose that for candidates  $A$  and  $B$ , constraint  $C_{fatal}$  is the fatal constraint, that is, the highest ranked constraint assigning either  $A$  or  $B$  an uncancelled mark following mark cancellation.*

*Then, the difference of the violation profiles of candidates  $A$  and  $B$  is the pair  $\langle C_{fatal}, C_{fatal}(A) - C_{fatal}(B) \rangle$ , that is: “the difference<sup>14</sup>  $C_{fatal}(A) - C_{fatal}(B)$  at constraint  $C_{fatal}$ ”.*

*Furthermore, the difference of the violation profiles is defined to be zero ( $\langle 0, 0 \rangle$ ) if the two candidates incur exactly the same violation marks (i.e. there is no uncancelled mark).*

For instance, in tableau (2.11), the difference of the violation profiles of  $w_1$  and  $w_2$  is the pair  $\langle C2, -2 \rangle$  (“−2 violations of C2”).

<sup>13</sup>Here the mark \*m is meant to be a token of violating constraint m.

<sup>14</sup>In linguistic applications, the levels of violation are non-negative integers, thus their difference is always an integer. Generalising to real-valued violation levels is straightforward. Problems may arise, however, if the constraints map to a fully ordered set in which subtraction is not defined. Yet, we do not deal with this case.

Let us introduce this definition now in a slightly more formal way. The point-wise difference of two profiles seen as vectors (equation (2.10)) is still a *violation profile-like vector*:

$$H(w') - H(w) = (C_N(w') - C_N(w), \dots, C_0(w') - C_0(w)) \quad (2.12)$$

This difference, however, does not have yet a form that can be used in (2.8).

When one compares two candidates, what matters according to the *Cancellation/Domination Lemma* is the leftmost non-zero component in this difference vector. This is the component corresponding to the fatal constraint: the highest ranked constraint that assigns a different number of marks to the two candidates. In standard OT, only its *sign* (positive or negative) matters. The crucial step in the present proposal is to take its *value* (as opposed to only its sign) and its *place* in the vector, but no further information.

Consequently, two difference vectors are *equivalent* for the present purpose if their leftmost non-zero component is the same and in the same column.<sup>15</sup>

**Definition 2.2.2.** *Two vectors,  $\underline{a} = (a_N, \dots, a_0)$  and  $\underline{b} = (b_N, \dots, b_0)$  are equivalent*

$$\underline{a} \cong \underline{b}$$

*if and only if there is a  $k \in \{N, \dots, 0\}$  such that  $a_k$  is the leftmost non-zero component of  $\underline{a}$ ,  $b_k$  is the leftmost non-zero component of  $\underline{b}$ , and  $a_k = b_k$ .*

*Additionally,  $(0, 0, \dots, 0) \cong (0, 0, \dots, 0)$ .*

In other words, we require both vectors to begin with the same number of zeros, and their first non-zero element to be also equal. They may differ in their further components.

It can be easily shown that  $\cong$  is indeed an *equivalence relation*:<sup>16</sup> it is reflexive ( $\underline{a} \cong \underline{a}$ ), symmetric ( $\underline{a} \cong \underline{b}$  implies  $\underline{b} \cong \underline{a}$ ) and transitive (if  $\underline{a} \cong \underline{b}$  and  $\underline{b} \cong \underline{c}$  then  $\underline{a} \cong \underline{c}$ ). Consequently,  $\cong$  defines *equivalence classes* on the set of the vectors:  $\underline{a}$  and  $\underline{b}$  belong to the same equivalence class iff  $\underline{a} \cong \underline{b}$ . The equivalence classes are disjoint (their intersection is empty) and cover the whole set of vectors. An equivalence class can be specified by the index of the left-most (that is, highest ranked) component, as well as by the value of this component.

What really interests us is not the difference as defined in Eq. (2.12), but the equivalence class to which this difference belongs. The philosophy of standard Optimality Theory, namely the *Cancellation/Domination Lemma*, does not differentiate between two difference vectors that belong to the same equivalence class with respect to equivalence relation  $\cong$ .

This is why we define the *magnitude* of a violation profile-like vector as the equivalence class to which the vector belongs:

**Definition 2.2.3.** *The magnitude of a violation profile-like vector  $(a_N, \dots, a_0)$  is*

$$\|(a_N, \dots, a_0)\| = \langle k, a_k \rangle$$

<sup>15</sup>This definition introduces the equivalence  $\cong$  of two vectors in general. This relation will be used on vectors representing the *difference* of two violation profiles. It is not to be confused with the equivalence relation in Definition 3.1.6, according to which two violation profile-like vectors are equal ( $H(w_1) = H(w_2)$ )—the candidates are equivalent ( $w_1 \simeq w_2$ )—iff they incur the same number of violation marks by each of the constraints.

<sup>16</sup>Cf. e.g. Eric W. Weisstein. "Equivalence Relation." From MathWorld—A Wolfram Web Resource. <http://mathworld.wolfram.com/EquivalenceRelation.html>

	$C_N$	$C_{N-1}$	...	$C_{k+1}$	$C_k$	$C_{k-1}$	$C_{k-2}$	...
$w'$	2	0		1	2	3	0	
$w$	2	0		1	3	1	2	
$H(w') - H(w)$	0	0		0	-1	2	-2	
$\ H(w') - H(w)\ $	0	0		0	-1	-	-	

Table 2.3: **An example for the difference of two violation profiles:** given the profiles of  $w$  and  $w'$ ,  $|w' - w| = \|H(w') - H(w)\| = \langle C_k, -1 \rangle$ , and  $C_k$  is the fatal constraint. The differences in the violation levels of the constraints ranked lower than  $C_k$  are ignored in the magnitude of a vector.

where  $k$  is the lowest (rightmost) element of  $\{N, \dots, 0\}$  such that  $\forall j > k$ : if  $j \leq N$  then  $a_j = 0$

Thus,  $\langle k, a_k \rangle$  is the name of the equivalence class to which all vectors belong whose first components (with indices  $N, N-1, \dots, k+1$ ) are zero, and whose  $k$ th component is  $a_k$ . If  $k = N$ , there are no zero components on the left.

Then, we define the *difference of two violation profiles* as simply the magnitude of the difference of the violation profile vectors:

**Definition 2.2.4.**

$$|w' - w| = \|H(w') - H(w)\| = \langle k, C_k(w') - C_k(w) \rangle$$

Here,  $C_k$  is the fatal constraint, that is, the highest ranked constraint with uncanceled violation marks (the highest  $k$  such that  $C_k(w') - C_k(w) \neq 0$ ):

$$\begin{aligned} \|H(w') - H(w)\| &= \|(0, 0, \dots, C_k(w') - C_k(w), \dots, C_0(w') - C_0(w))\| = \\ &= \langle k, C_k(w') - C_k(w) \rangle \end{aligned} \quad (2.13)$$

An example is given in Table 2.3.

Note that the difference of two violation profiles is a pair of *numbers*: the first number is the index of a constraint, whereas the second number is a difference in violation levels. So far, both have been integers, but later we shall be more flexible.

We have already introduced the *Law of trichotomy* in Chapter 1, and we shall return to it soon. It states that for any two violation profiles  $A$  and  $B$ , exactly one of the following statements holds: 1.  $A$  is better than  $B$  ( $A \succ B$ ); 2.  $A$  is equivalent to  $B$  (same violation profile,  $A \simeq B$ ); 3. and  $A$  is worse than  $B$  (that is,  $A \prec B$ ). By using the *Cancellation/Domination Lemma*, it is easy to demonstrate that these three cases correspond to the second component in the difference of the violation profiles being negative, zero or positive, respectively. Consequently, it is well-founded to represent *violation profiles* by *violation vectors*. Section 3 will introduce two further ways of representing violation profiles, prove their well-foundedness, and derive ways of applying simulated annealing to Optimality Theory.

Even if we could not reduce the difference of two violation profiles, the expression  $H(w') - H(w)$  appearing in pseudo-definition (2.8), to a real value, we have now a pair of numbers instead of an  $N$ -dimensional vector.

We shall soon need—for the introduction of temperature—the comparison of two such pairs. Which jump is greater: moving from  $w_1$  to  $w_2$ , or from  $w_3$  to  $w_4$ ? Increasing the violation level of a higher ranked constraint is a steeper step than increasing the number of marks assigned by lower ranked constraints only. If both of two steps increase the violation level of constraint  $C_k$  (without increasing the violation level of higher ranked ones, and we do not care about lower ranked ones), then the step that adds more violation marks is the steeper one, although the steepness of the two steps do not differ dramatically.

Consequently, we propose the following relations (not all of which exclude the other ones):

**Definition 2.2.5.** *Let  $K_1$  and  $K_2$  be real numbers, while  $t_1$  and  $t_2$  be positive real numbers.*

1. *Two pairs are equal:*  
 $\langle K_1, t_1 \rangle = \langle K_2, t_2 \rangle$ , iff  $K_1 = K_2$  and  $t_1 = t_2$ .
2. *One of the pairs is greater:*  
 $\langle K_1, t_1 \rangle > \langle K_2, t_2 \rangle$ , iff either  $K_1 > K_2$  or  $K_1 = K_2$  and  $t_1 > t_2$ .
3. *One of the pairs is much greater:*  
 $\langle K_1, t_1 \rangle \gg \langle K_2, t_2 \rangle$ , iff  $K_1 > K_2$ .
4. *Two pairs are approximately equal:*  
 $\langle K_1, t_1 \rangle \approx \langle K_2, t_2 \rangle$ , iff  $K_1 = K_2$ .

After this slight detour, let us turn back to our agenda set up earlier. We have argued for a definition of the difference of two violation profiles. How does it help us in defining temperature?

### Defining temperature and the transition probabilities

Recall the slight move of replacing the fatal constraint by its index in Definition 2.2.3: we shall use  $\langle k, C_k(w') - C_k(w) \rangle$ , and not  $\langle C_k, C_k(w') - C_k(w) \rangle$ . The goal of this slight change has been to help defining temperature.

As discussed earlier, the “temperature” in simulated annealing determines the range of change in energy (harmony, or some other function to be optimised), above which counter-optimal moves are prohibited, and under which counter-optimal moves are allowed. Therefore, temperature has to have the same “form” (dimension, structure) as changes in the function to be optimised. As a difference of two violation profiles is now a pair  $\langle k, t \rangle$ , we define temperature also as a pair of numbers:

$$T = \langle K_T, t \rangle \in \mathcal{R} \times \mathcal{R}^+ \quad (2.14)$$

If  $K_T = i$  such that there is a constraint  $C_i$ , then the temperature  $T$  is said to be *in the domain of* constraint  $C_i$ . The temperature  $T = \langle K_T, t \rangle$  is *above* (*below*) constraint  $C_i$  if  $K_T > i$  ( $K_T < i$ ). Using Definition 2.2.5, temperature is *in the domain of* constraint  $C_i$  iff  $T \approx \langle i, 1 \rangle$ ; and temperature *above* the domain of constraint  $C_i$  iff  $T \gg \langle i, 1 \rangle$ . Temperature being *far above* the domain of  $C_i$  (if one wishes,  $T \gg \gg \langle i, 1 \rangle$ ) will denote  $K_T \gg i$ , in an informal sense.

$t \leq 0$  is not allowed by definition, because, as we shall see, that would lead to mathematical problems in the simulation. The situation is similar to the

one in physics, where zero and negative absolute temperatures (Kelvin's scale) are also prohibited. On the other hand,  $K_T$  can be both positive and negative, and behaves like relative temperature scales in physics (Celsius, Fahrenheit, Réaumur). In practice,  $K_T$  will be an integer, although nothing prohibits having non-integer values for  $K_T$ . In addition,  $K_T$  will range between  $K_{max}$ , usually (far) above the highest ranked constraint, and  $K_{min}$ , always far below the lowest ranked constraint. Assigning index 0 to the lowest ranked constraint causes the system to freeze exactly when the first component of the temperature  $K_T < 0$ , as the continental reader might expect.<sup>17</sup>

Notice that if  $K_T$  is the index of some constraint—and this case will be the most interesting one—, then temperature  $\langle K_T, t \rangle$  corresponds to some equivalence class on the set of possible violation profile vector differences. Namely, to the vector differences whose first (leftmost) components (with indices  $N, N - 1, \dots, K_T + 1$ ) are zero, and whose  $K_T$ th component is  $t$  (by Definition 2.2.3). In other words, such a temperature is equivalent to a move which increases the number of violation marks assigned by constraint  $C_{K_T}$  by  $t$ , leaves the violation marks of higher ranked constraints unchanged, and might change the violation marks assigned by lower ranked constraints in any way.

However,  $K_T$  does not necessarily correspond to some constraint. This is why the structure of temperature is said to be a *generalisation* of the violation profile differences. And yet, we can also apply the comparison relations introduced by Definition 2.2.5 to such generalisations. Using this definition, a *decreasing series of temperature values* (a *cooling schedule*) can be specified. Moreover, it is possible to compare a change in the harmony function to the actual temperature, in order to define the transition probabilities.

Remember Table 2.2, based on equation (2.7): temperature in simulated annealing draws a smooth border line between allowed and prohibited counter-optimal transitions. The chance of increasing the energy function with  $T$  is  $1/e$  at temperature  $T$ .

Combining Definition 2.2.5 with Table 2.2, we can easily formulate now the definition of the transition probability for the  $w \succ w'$  case:

$$P(w \rightarrow w'|T) = \begin{cases} 1 & \text{if } \|H(w') - H(w)\| \ll T \\ 1/e & \text{if } \|H(w') - H(w)\| = T \\ 0 & \text{if } \|H(w') - H(w)\| \gg T \end{cases} \quad (2.15)$$

In words, if temperature is in a domain above that of the fatal constraint, then the loss of Harmony (the increase in violation marks) incurred by the move is negligible compared to the temperature, and the move is always taken. If, on the other hand, the temperature is very cold compared to the increase in violation marks, then the move is never taken.

According to Table 2.2, if the increase in the cost function is comparable to the temperature, then the probability of moving has a medium value. Specifically, if the two are equal, the move has a chance of  $1/e$ . By Definition 2.2.5, this translates to the case when the temperature is exactly in the domain of the fatal constraint. In this special case, equation (2.7) can be simply copied: augmenting the violation marks of the fatal constraint by  $d$  has a probability of  $e^{-d/t}$ , where  $t$  is the second component of the temperature in equation (2.14).

<sup>17</sup>Even if weird, nothing prohibits the Anglo-Saxon reader from assigning index 32 to the lowest ranked constraint.

In particular, if  $\|H(w') - H(w)\| = T$ , that is,  $d = t$ , the transition probability is indeed  $e^{-1} = 1/e$ .

In summary, the transition probabilities for the  $w \succ w'$  case are:

$$P(w \rightarrow w'|T) = \begin{cases} 1 & \text{if } \|H(w') - H(w)\| \ll T \\ e^{-d/t} & \text{if } \|H(w') - H(w)\| \approx T \\ 0 & \text{if } \|H(w') - H(w)\| \gg T \end{cases} \quad (2.16)$$

where  $d$  is the second component of  $\|H(w') - H(w)\|$ , and  $t$  is the second component of  $T$ .

Equation (2.16) differs from the behaviour of the traditional transition probabilities summarised in Table 2.2 only in one respect. Namely, in traditional simulated annealing (and in physics), the  $\Delta E \gg T$  and  $\Delta E \ll T$  cases were informal notions, and consequently, the statements  $P(w \rightarrow w'|T) \approx 0$  and  $P(w \rightarrow w'|T) \approx 1$  were also to be taken informally. SA-OT, however, implements a non-real valued optimisation (due to the Strict Domination Hypothesis of OT), therefore the  $\gg$  relation could be introduced exactly in Definition 2.2.5 (again, due to the Strict Domination Hypothesis of OT). This is the reason why the probabilities in the  $\Delta H \gg T$  and the  $\Delta H \ll T$  cases are postulated to be exactly 0 and 1, respectively.

At this point, we are already getting very close to the introduction of the SA-OT algorithm. We have defined temperature and the transition probabilities. The very last step is to introduce the cooling schedule, that is, a series of decreasing temperature values. The way to do this is already implicit in Definition 2.2.5.

The cooling schedule is realised in standard simulated annealing as a single loop gradually decreasing the temperature (Fig. 2.2). Thereby, the probability of jumps increasing the cost function are gradually decreased from very close to 1 to very close to 0. At each moment, temperature defines the jump that has a probability of  $1/e$ .

What should the cooling schedule do in SA-OT? Initially, it should allow any transitions. Then, it should prohibit transitions increasing the violation level of highly ranked constraints. Then, prohibit also the transitions that would only increase the violation marks assigned by lower ranked constraints, etc. In the final phase of the simulation, no move to a worse state should be allowed.

According to equation (2.16), a temperature that would allow any move has a first component  $K_T$  that is higher than the index of the highest ranked constraint. A temperature that prohibits augmenting the violation level of the highest ranked constraint, but allows augmenting the violations of lower ranked constraints, has a  $K_T$  value that is lower than the index of the highest ranked constraint, but higher than the indices of the lower ranked constraints. Finally, if  $K_T$  is lower than the index of the lowest ranked constraint, no transition to a worse candidate is allowed. Consequently, we have to diminish  $K_T$ .

Furthermore, we also want to diminish  $t$  in a more fine-grained way. At a given point,  $C_{K_T}$  is the highest ranked constraint whose violation marks can increase. Still, equation (2.16) prefers increasing the number of these violation marks by  $d = 1$  over by  $d = 2$ . If initially,  $t = 5$ , taking both steps are relatively easy. As  $t$  is decreased, the chances of both moves are smoothly turned off, although the  $d = 2$  jump will be always less likely than the  $d = 1$  one.

In sum, we add an embedded loop diminishing  $t$  into the loop diminishing  $K_T$ . Such a double loop will mimic traditional simulated annealing. At each time of the simulation, temperature shows what is the jump that has a probability of  $1/e$ . Steeper jumps are less probable, and much steeper jumps have a zero probability. Smaller jumps have a higher probability, and much smaller jumps have a probability of 1. If the value of temperature corresponds to some equivalence class on the set of profile differences, its meaning can be simply translated as “increase the violation marks assigned by constraint  $C_{K_T}$  by  $t$ , do not change the violation marks assigned by higher ranked constraints, and do what you want with the violation levels of lower ranked constraints”. Otherwise, the temperature cannot be translated into a change in the violation profile, but can be compared to such changes, by using Definition 2.2.5.

The following picture may help to visualise the idea better. Take a scale which is composed of intervals called *domains*, denoted by  $K = 5, K = 4, \dots, K = 0, K = -1, \dots$ <sup>18</sup>

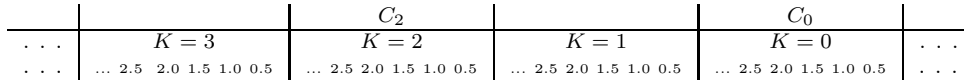


Figure 2.7: Visualising the domains traversed by temperature

Each domain is an interval open on the left and closed on the right, so that the interval  $(+\infty, 0]$  can be projected onto it. Temperature  $T = \langle K, t \rangle$  is represented on this scale by the point that is the projection of  $t$  onto the domain  $K$ . Moving to the right on this scale means decreasing temperature: remember,  $T_1 = \langle K_1, t_1 \rangle > T_2 = \langle K_2, t_2 \rangle$  iff either  $K_1 > K_2$ , or  $K_1 = K_2$  and  $t_1 > t_2$ .

Furthermore, some of the domains correspond to constraints—in the present example, constraint  $C_2$  corresponds to  $K = 2$ , and constraint  $C_0$  corresponds to  $K = 0$ . The higher ranked a constraint, the higher the domain  $K$  it is associated with (in the present case, thus,  $C_2 \gg C_0$ ). Notice that here we already enjoy the advantages of the initially surprising notation that associates higher indices with higher ranked constraints. Temperature may be assigned values that are in domains corresponding to some constraints, or values that are in domains above / between / below some constraints.<sup>19</sup>

Now, decreasing temperature can be realised by decreasing  $t$  within one domain, and then jumping to the next domain (or to another domain further below). In practice, embedded loops will be used: the outer cycle decreases the domain of the temperature ( $K$  descends from  $K_{max}$  to  $K_{min}$ , by steps  $K_{step}$ ), and the inner loop decreases  $t$  within a given domain (from  $t_{max}$  to  $t_{min}$ , by steps  $t_{step}$ ). Importantly, the borders of the domains are taboo:  $t > 0$  always, similarly to absolute temperature in physics.

<sup>18</sup>This approach, similarly to the applications to follow, supposes that the first component of the temperature  $T = \langle K, t \rangle$  may take only integer values.

<sup>19</sup>In what follows, we shall place constraints into the domains  $K = 0, K = 1, K = 2$ , etc. Nothing prohibits us, however, from leaving out some domains. Then, in some phase of the simulation, temperature may take values from the domain lying between the domains associated to two successive constraints.

### 2.2.4 Introducing the SA-0T algorithm

At this point we are able to formulate the way simulated annealing will be implemented for Optimality Theory. We will soon introduce the precise algorithm for *Simulated Annealing Optimality Theory* (Fig. 2.8).

We begin the random walk in the space of the candidates by choosing (randomly or semi-randomly) an initial candidate  $w_0$  (recall Fig. 2.5). In the case of a finite candidate set, we shall use each candidate as a starting point with equal probability. In practice, we shall run the simulation many times from each of the candidates. In the case of an infinite candidate set, however, a more useful solution is to start the simulation with equal probability from the elements of a small finite subset: for instance, from the candidates without the recursive insertion that results in an infinite candidate set. A third option is to launch the simulation always from the candidate that is arguably the default one with respect to the underlying representation.<sup>20</sup>

In the beginning, temperature  $T_0 = \langle K_{max}, t_{max} \rangle$  is high, that is,  $K_{max}$  is higher than the domain of the highest ranked constraint. A lower initial temperature can also be chosen, but if  $K_{max}$  is higher than the index of the highest ranked constraint, the simulation may have an initial phase in which the transition probability  $P(w \rightarrow w')$  is 1 for all  $w$  and  $w'$ . The advantage (or disadvantage, depending on what your goal is) of setting  $K_{max}$  high enough is to reduce the influence of the initial candidate's choice.

At each time step of the simulation, temperature is decreased, and one step may be performed in the search space. Traditional simulated annealing sometimes allows for more steps before reducing the temperature (cf. the parameter  $nrep$  in Fig. 2.2). Instead of introducing an additional parameter, we rather set  $nrep = 1$ , and prefer to reduce the temperature in smaller steps. The difference between the two approaches should not be significant, but a proliferation of parameters might render our analysis more complex. Nonetheless, further research could analyse the role of this factor, as well.

At a given moment in the simulation, the random walker is located in the position represented by candidate  $w$ . Temperature then is  $T = \langle K_T, t \rangle$ . A neighbour  $w'$  of candidate  $w$  (cf. Fig. 2.5) is randomly picked. The choice is determined by the *topology* of the search space: the neighbourhood structure provides the set  $Neighbours(w)$ , from which  $w'$  is chosen using the *a priori* probability distribution  $P_{choice}(w'|w)$  on  $Neighbours(w)$ . As discussed in section 2.2.2, the elements of  $Neighbours(w)$  may have equal or different chance to be picked, and several strategies exist to define  $Neighbours(w)$ .

There follows a comparison of the violation profiles of  $w$  and  $w'$ . If  $w'$  is more harmonic than  $w$  ( $w' \succ w$ ), or they are equally harmonic ( $w' \simeq w$ : they incur exactly the same violation marks), the random walker automatically moves to  $w'$ . Otherwise, moving to  $w'$  depends on the temperature. How? The likelihood is defined in equation (2.16).

To sum up, if  $T = \langle K_T, t \rangle$  and  $\|H(w') - H(w)\| = \langle k, d \rangle$ , then the transition probability is:

---

<sup>20</sup>Suppose for instance that the stress pattern of a compound word has to be predicted. Then the default candidate could be the one whose stress pattern is the concatenation of the stress patterns of the two parts of the compound. We shall return to this idea in section 5.7.

$$P(w \rightarrow w' | T) = \begin{cases} 1 & \text{if } d \leq 0 \\ 1 & \text{else if } k < K_T \\ e^{-d/t} & \text{else if } k = K_T \\ 0 & \text{else} \end{cases} \quad (2.17)$$

We can reformulate this equation in words:

**RULES OF MOVING:** Let the crucial constraint (the highest ranked constraint with uncanceled marks) when comparing  $w$  to  $w'$  be  $C_k$ , and temperature be  $T = \langle K_T, t \rangle$ . Then the following options are available:

- If  $w'$  is better than  $w$  ( $w' \succ w$ , that is,  $C_k(w') < C_k(w)$ ), then move from  $w$  to  $w'$ .
- If  $w'$  loses due to the critical constraint  $C_k > K_T$ : don't move!
- If  $w'$  loses due to the critical constraint  $C_k < K_T$ : move!
- If  $w'$  loses due to the critical constraint  $C_k = K_T$ : move with probability  $P(w \rightarrow w') = e^{-d/t}$ , where  $d = C_k(w') - C_k(w)$ .

The case  $k < K_T$  corresponds to  $\Delta E \ll T$  in standard simulated annealing: the transition is highly probable. Similarly,  $k > K_T$  means in the OT philosophy that  $\Delta E \gg T$ , and the transition is prohibited. The middle case,  $\Delta E \approx T$ , corresponds here to  $k = K_T$ , and the exponential function ensures a smooth transition between the two extremes. In this last case, as temperature and the violation profile difference are in the same domain, the second (real-valued) components of both play the main role. Otherwise, these second components— $t$  and  $d$ —do not get to the stage.

Temperature  $T = \langle K_T, t \rangle$  is decreased in a double loop. The outer one diminishes  $K_T$  from  $K_{max}$  to  $K_{min}$ , in linear steps of  $K_{step}$ . Similarly, the inner cycle reduces  $t$  from  $t_{max}$  to  $t_{min}$ , in linear steps of  $t_{step}$ . The linearity of the outer cycle follows directly from the picture presented in Fig. 2.7.

However, the component  $t$  of the temperature could be decreased in several other ways, as well, for instance on a logarithmic scale.<sup>21</sup> Still, according to the literature on standard simulated annealing Reeves (1995), the exact way of decreasing temperature does not have a major influence on the precision. Consequently, we opt for the simplest way, and leave alternatives for future research. Section 5 will examine the role of  $t_{max}$  and  $t_{min}$ , and will conclude that different ways of running the inner loop may result in minor—though statistically significant—results. Thus, I suppose that a logarithmic scale would have similar consequences.

The algorithm in Fig. 2.2 also includes a *stopping condition*. Frequently, a simulated *specific heat* is measured, that is, the improvement in energy (in the cost function) per unity temperature change ( $\partial E / \partial T$ ). The stopping condition then requires this measure to drop below a certain level, that is, the algorithm terminates when decreasing the temperature does not lead to much decrease in the cost function anymore.

<sup>21</sup>A logarithmic scale involves multiplying  $t$  each time with a constant factor smaller than 1:  $t_{n+1} = t_{step} \cdot t_n$ . Whereas a linear scale adds a constant negative value to it:  $t_{n+1} = t_n - t_{step}$ .

```

ALGORITHM: Simulated Annealing for Optimality Theory
Parameters: w_init, K_max, K_min, K_step, t_max, t_min, t_step
           # t_step: number of iterations / speed of simulation
w <-- w_init ;
  for K = K_max to K_min step K_step
    for t = t_max to t_min step t_step
      choose random w' in neighbourhood(w) ;
      calculate < C , d > = ||H(w')-H(w)|| ;
      if d <= 0 then w <-- w'
      else
        w <-- w' with probability
          P(C,d;K,t) = 1      , if C < K
                    = exp(-d/t) , if C = K
                    = 0      , if C > K
    end-for
  end-for
return w

```

Figure 2.8: **The algorithm of Optimality Theory Simulated Annealing:** Moving with probability  $P(C,d;K,t)$  is a short-hand for generating a random number  $0 \leq r \leq 1$ , and moving iff  $r \leq P(C,d;K,t)$ .

We could have included a similar condition: the algorithm stops whenever the random walker has not moved for a certain number of time steps. For instance, if the random walker has not left candidate  $w$  for  $c \cdot |\text{Neighbours}(w)|$  steps (with  $c = 5$ , say), we may safely conclude that  $w$  is a local optimum, and temperature has become too cold to be able to escape it. Running further the algorithm makes no sense, for the likelihood of such an escape will further drop with decreasing temperature. If  $c$  is set too low, the algorithm may frequently stop in candidates that are not local optima, but have a few neighbours that are even worse. If  $c$  is set too high, the simulation may run unnecessarily long. Nonetheless, it is not difficult to come up with a reasonable compromise.<sup>22</sup> The danger only arises in the case of a search space that has “locally optimal valleys” formed by neighbours that incur exactly the same violation marks and are more harmonic than their environment. Such a system is prone to run into an infinite loop, so one either has to employ also a  $K_{min}$  limit, or to count horizontal moves as if the system were stuck in a state (that is, count time steps without improvements in the Harmony function).

The algorithm of *Optimality Theory Simulated Annealing* (OT-SA) is finally presented in Fig. 2.8 in the form we shall use it.

<sup>22</sup>Suppose that a candidate has  $n$  neighbours, only one of which is more harmonic than this candidate. Temperature is low enough, so that the random walker cannot move to a less harmonic neighbour. Suppose, furthermore, that each neighbour has equal *a priori* probability. Then, the likelihood of choosing a neighbour to which the system will not move is  $\frac{n-1}{n}$ . Trying  $c \cdot n$  times, and still not leaving this non-local optimum has a probability of

$$\left(\frac{n-1}{n}\right)^{cn} < e^{-c}$$

because the series  $(1 + \frac{1}{n-1})^n$  is a monotonically decreasing series with  $e$  as its limit. Consequently,  $c = 3$  guarantees you to terminate the algorithm in a local optimum with probability 95%, and  $c = 5$  with a probability higher than 99%. Moreover, this likelihood is even much higher in most search spaces.

Compare it to the standard simulated annealing algorithm in Fig. 2.2 in section 2.1.2. The two main changes are the way temperature is decreased and the transition probability is calculated. Both are connected to the more complex, non-real valued character of the function to be optimised (the cost function). Temperature is decreased in a double cycle, as explained when temperature was introduced: the outer loop decreases its first component and the inner loop its second component. Furthermore, the transition probability can also be determined in a more complex way, which follows equation (2.17).

As can be seen, the parameters of the algorithm are the initial candidate ( $w_0$ ) from which the simulation is launched, as well as the parameters of the cooling schedule:  $K_{max}$ ,  $K_{min}$ ,  $K_{step}$ ,  $t_{max}$ ,  $t_{min}$ ,  $t_{step}$ .

Typically,  $K_{max}$  will be higher than the highest ranked constraint, so that there will be a phase of the simulation when the random walker can freely increase the violation marks of even the highest ranked constraints. One may however add constraints higher ranked than  $K_{max}$ : the resulting picture will be as if we restricted GEN, since the candidates sub-harmonic for these “hyper-strong” constraints would not play a role at all in the model. Section 6.5 introduces a model whose success depends on tuning  $K_{max}$ , a parameter that does not influence the model in Chapter 5 (other than trivially).

If the role of  $K_{max}$  is to supply additional layers above the highest ranked constraint in order to allow the random walker to move unhindered in the initial phase of the simulation, then the role of  $K_{min}$  is to define the length of the final phase of the simulation. Namely, by having  $K_{min}$  (much) below the lowest ranked constraint, the system is given enough time to “relax”, to reach the closest local optimum, that is the bottom of the valley (the top of the hill, if you maximise the function) in which the system is stuck. Without such a final phase, the system will return any candidate, not only local optima, leading to an uninteresting model. Consequently,  $K_{min}$  has to be chosen such that the number of iterations in this frozen state ( $K_T < 0$ , if the lowest ranked constraint is  $C_0$ ) be enough to reach the closest local optimum by a hindered random walk.<sup>23</sup>

Not much attention will be given to the parameter  $K_{step}$ . Although other options are also possible, the standard way we shall proceed is the following: if we have  $N + 1$  constraints, we always place them into the domains  $K = 0$ ,  $K = 1, \dots$ ,  $K = N$ , and we set  $K_{step} = 1$ . Further, if not specified otherwise,  $K_{max} = N + 1$ . Moreover, the system is said to be *frozen* if the temperature  $T$  drops below domain  $K = 0$ , that is,  $T = \langle K_T, t \rangle$  with  $K_T < 0$ . In a frozen system, no move can lead to a less harmonic candidate.

The parameters  $t_{max}$ ,  $t_{min}$  and  $t_{step}$  drive the inner loop of the algorithm, that is the decreasing of the second component  $t$  of temperature  $T = \langle K, t \rangle$ . This second component plays a role only in the expression  $e^{-d/t}$ , used when the crucial constraint at which  $w'$  is defeated by  $w$  coincides with the domain of the current temperature. Because the neighbouring candidates  $w$  and  $w'$  typically differ only minimally (a *basic operation* transforms  $w$  into  $w'$ ), their violation profile is also quite similar, thus the difference  $d$  in violating the crucial constraint is expected to be a low number (usually  $1 \leq |d| \leq 2$ ). Consequently, the  $e^{-d/t}$  vanishes if  $t \gg 3$ , and so the default values used will be  $t_{max} = 3$ ,

<sup>23</sup>In our models—with  $T_{step} = 1$ —a good choice might be  $K_{min} = -100$ . If  $T_{step} = 0.01$ , however,  $K_{min} = -1$  is enough, and  $K_{min} = -100$  will make the simulation unnecessarily long. Remember that the constraints are located in the domains  $k = 0, \dots, N$ . My scripts automatically adjust  $K_{min}$  to  $T_{step}$ .

$t_{min} = 0$ .<sup>24</sup>

Chapter 5 will also present experiments tuning  $t_{min}$  and  $t_{max}$ . Increasing or decreasing the half-closed interval  $[t_{min}, t_{max})$  crossed by  $t$  has a measurable, though minor effect on the outcome of the simulation. Based on this result, we argue that  $t$  and the exponential expression  $e^{-d/t}$  in the definition of SA-OT does have an effect on the outcome of the simulation.

The most interesting parameter is  $t_{step}$ , for it is inversely proportional to the number of iterations performed—if the other parameters are kept unchanged. In this way, it directly controls the speed of the simulation, hence, indirectly, its precision. Therefore, most of our experiments will vary this parameter. Actually, the other parameters also may change the number of iterations performed, but their effect is more complex, so tuning  $t_{step}$  is the most straightforward way—at least, for me—to change the speed. On the other hand, when measuring the role of  $t_{max}$  and  $t_{min}$  in Chapter 5, the parameter  $t_{step}$  will help in controlling for the number of iterations.

In the next chapter, additional arguments are brought in favour of the SA-OT Algorithm in Fig. 2.8: different formal approaches will lead to the same proposal. Before that, however, let us perform some first experiments using the *Simulated Annealing Optimality Theory Algorithm* in order to get an impression of it.

## 2.3 Playing with SA-OT

### 2.3.1 When SA-OT works

We have defined the *Simulated Annealing for Optimality Theory Algorithm* (SA-OT), so it is now high time to try out whether it really works, and to see under what circumstances it “fails”.

In the present section, we are trying out toy models in order to obtain a better understanding of what SA-OT really does in practice. The reader is welcome to implement these examples personally on the demo SA-OT available on my web site at <http://www.let.rug.nl/~birot/sa-ot/>. One can simply vary the different parameters of the algorithm, and examine its behaviour. Furthermore, one can specify a “verbose” output, which explains all the details during the simulation.

In section 2.1.2, we introduced a search space with three states and with an asymmetric lambda-shape neighbourhood structure (Fig. 2.3) in order to show why traditional simulated annealing works. Now, we shall try out analogous cases in Optimality Theory and analyse the performance of SA-OT.

Among the three candidates, B is a neighbour of A and C, whereas A and C have the singleton set {B} as their neighbour set. In other words, A and C are not neighbours of each other. Candidate C is always the global optimum, to which the usual  $\blacksquare$  symbol points, whereas candidate A is a local optimum, annotated by the variation symbol  $\sim$  (Fig. 2.9). If A and C were neighbours, A could not become a local optimum.

<sup>24</sup>Notice that for the sake of convenience,  $t_{min}$  is always understood as the lower border of  $t$ , *exclusively*. The inner loop runs while  $t > t_{min}$ . Nevertheless,  $K_{min}$  will be understood as the lower border, *inclusively*: the outer loop is meant to run while  $K \geq K_{min}$ —whenever rarely  $K_{min}$  will be mentioned. This slight incongruity should render the notations otherwise simpler.

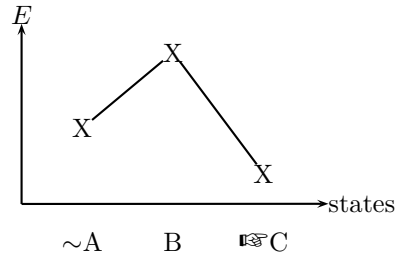


Figure 2.9: An asymmetric landscape with three states, two of which are local optima, but only state C is a global optimum. State B is a neighbour of both A and C, however states A and C are not neighbours of each other.

The first tableau to be examined is the following:

		$C1$
$\sim$	A	*
	B	**
$\text{☞}$	C	

(2.18)

It is easy to check that  $C \succ A \succ B$ . The only constraint  $C1$  is assigned index (a  $K$ -value) of 0, that is, temperature will be in its domain whenever the first component of  $T$  is  $K_T = 0$ . Let us set  $K_{max} = 1$ ,  $K_{step} = 1$ ,  $t_{max} = 3$ ,  $t_{min} = 0$ , and  $K_{min}$  low enough, say,  $K_{min} = -100$ . The only parameter we vary is  $t_{step}$ .

As  $K_{max}$  is higher than the rank of the only constraint, the initial candidate of the simulation will not really matter. In practice, however, we launch the simulation in 1/3 of the cases with A as the initial candidate, in 1/3 of the cases with B as the initial candidate, and in 1/3 of the cases with C as the initial candidate.

The algorithm is stochastic, and is prone to return different outputs. If  $K_{min}$  were higher than the rank of the constraint, even B would be returned in 1/3 or 2/3 of the cases, depending on the parity of the number of the iterations. With different parameter settings, the algorithm terminates only in A and C, or exclusively in C. Therefore, what will interest us is not the output of a certain simulation, but the *proportions* of the different outputs of several simulations. In other words, we seek to know the *likelihood* of the simulation to return a certain candidate—similarly, to other stochastic linguistic models referred to in chapter 1.<sup>25</sup>

The following table contains the absolute frequencies returned by a number of experiments performed on the demo web site, with the simulation launched 100 times from each of the three candidates:

<sup>25</sup>By running the simulation many times, the *proportion* of a certain output (the relative frequency) should approximate the theoretical *probability* of returning that candidate.

$t_{step}$	Frequency of A	Frequency of C
3	141	159
3	146	154
3	145	155
3	129	171
3	131	169
1	128	172
1	122	178
1	134	166
1	130	170
1	127	173
0.1	79	222
0.1	86	214
0.1	82	218
0.1	72	228
0.1	90	210
0.01	32	268
0.01	33	267
0.01	40	260
0.01	40	260
0.01	27	273
0.001	9	291
0.001	10	290
0.001	12	289
0.001	6	294
0.001	14	286

(2.19)

Based on these data, the probability and the standard deviation ( $n$ ) of returning the globally optimal candidate, C, in function of  $t_{step}$  is:

$t_{step}$	Frequency of C
3	$0.537 \pm 0.0237$
1	$0.573 \pm 0.0130$
0.1	$0.727 \pm 0.0207$
0.01	$0.883 \pm 0.0167$
0.001	$0.967 \pm 0.0087$

(2.20)

The results speak for themselves. A very fast cooling schedule, such as  $t_{step} = 3$ , returns the local—but non-global—optimum A in almost half of the cases. Slowing down the cooling schedule, that is, increasing the number of iterations performed, will increase the likelihood of returning the global optimum in a highly significant level. For  $t_{step} = 0.001$ , the chance of returning A is below 5%, and an even smaller  $t_{step}$  would further reduce the frequency of A. This case, in which the frequency of the global optimum increases gradually to 100% as  $t_{step}$  decreases, will be considered a success of SA-OT.

One can also check simply that the choice of the initial candidate does not influence significantly the simulation, if  $K_{max}$  is large enough. For instance, here are the absolute frequencies for the outputs, when we launched the simulation 500 times from each of the three candidates, with  $t_{step} = 1$  (the data in

parentheses are the results of repeating the experiment twice):<sup>26</sup>

Initial candidate	Returning A	Returning C
A	211 (218, 231)	289 (282, 269)
B	191 (198, 195)	309 (302, 305)
C	198 (227, 218)	302 (273, 282)

(2.21)

We can now proceed to cases where SA-OT does not work in the sense that decreasing  $t_{step}$  will not increase the likelihood of returning the globally optimal (that is, the grammatical) candidate. In such cases, the topology of the candidate set forces the simulation to always return local optima with a constant likelihood. However, empirical research might find phenomena that can be accounted for by these systematic failures, as we shall argue for in section 6.4. The main reason for these systematic failures will be that in our definition of violation profile difference we neglect constraints that are below the fatal constraint.

### 2.3.2 When SA-OT *does not* work

Remember that a crucial step in introducing SA-OT was neglecting the constraints below the *fatal constraint* (the highest ranked constraint with uncanceled marks). The difference of two violation profiles is the difference in their levels of violating the fatal constraint, independently of how they behave with respect to lower constraints. Consequently, the difference of  $w$  and  $w_1$  is the same as the difference of  $w$  and  $w_2$ —violation  $C_1$  once—in the following tableau:

	$C_2$	$C_1$	$C_0$
$w$	*	**	
$w_1$	*	*	
$w_2$	*	*	**

(2.22)

We argued for this definition based on the foundations of Optimality Theory, following the claim that cumulativity effects should be avoided. Nevertheless, neglecting lower ranked constraints—not making any difference between the difference of  $w$  and  $w_1$  on the one hand, and the difference of  $w$  and  $w_2$  on the other in tableau (2.22)—leads us to cases where the algorithm for OT-SA just presented does not work.

Imagine again the asymmetric lambda-shaped landscape with three candidates, two of which are local optima, such as the one presented in figure 2.9. This is the repetition of figure 2.3, which we used in section 2.1.2 to understand why traditional simulated annealing works.

Remember the argumentation there. Moving to both directions from the middle state B has equal chance: both neighbours are chosen with equal probability, and the random walker moves there certainly, once chosen. Yet, it is more difficult to escape from the global optimum C than from the local optimum A, because moving to B involves a greater step uphill. So a slower cooling schedule with more time steps  $n$  makes it more probable that you will escape from A,

<sup>26</sup>Notice that  $t_{step} = 1$  and  $K_{max} = 1$  allows only for three unhindered steps, namely when  $T = \langle 1, 3 \rangle$ ,  $T = \langle 1, 2 \rangle$  and  $T = \langle 1, 1 \rangle$ . Changing the parameter setting would increase the number of unhindered steps, and would therefore decrease further the role of choosing the initial candidate.

and at least once choose to go to C, where you will be caught.<sup>27</sup> In order not to get confined in C, you have to choose always moving to A from B, which has a probability of  $0.5^n$  vanishing with high  $ns$ . The chance of choosing C—and get stuck there—at least once in  $n$  time steps is  $1 - 0.5^n \approx 1$  for high  $ns$ , that is, for slow cooling schedules.

Consequently, although gradient descent would assign the same probability of ending in A and C, the stochastic process introduced by simulated annealing increases the chance of finding the global optimum. The slower the cooling schedule, the higher the chance to find it.

Can our proposed simulated annealing for Optimality Theory exhibit the same behaviour? It depends on the profile of the three candidates, A, B and C. Without changing the topology, let us observe the behaviour of OT systems with different violation profiles. The simplest example was the one analysed in the previous subsection, namely, tableau (2.18), and there, simulated annealing worked perfectly. The effect just observed in the real-valued case also works if the three candidates are characterised by the following tableau:

	C1	C2
~ A	*	
B	*	*
☞ C		

(2.23)

Using the demo web site at <http://www.let.rug.nl/~birot/sa-ot/>, we can observe that finding the global optimum—to which the hand points—is even easier in this system than in the one defined by tableau (2.18). Namely, at  $t_{step} = 0.5$ , the likelihood of terminating in candidate A—the alternative form marked with the  $\sim$  symbol—is already below 10%; whereas  $t_{step} = 0.1$  will return exclusively the globally optimal candidate C.

What happens in this case? When temperature  $T$  is in the range of the constraint  $C2$ , moving from C to B is already impossible, for such a move would increase the number of violations of constraint  $C1 \gg T$ . However, escaping from A to B is still possible, and thus a slower schedule will result in a higher probability of escaping from A and falling into the trap of C. The more time steps are allowed while  $T$  is in the domain of  $C2$ , the higher the probability of ending up in C.

Thus, tableau (2.23) represents a second case where SA-OT behaves the same way as traditional simulated annealing. Take now the following tableau:

	C1	C2
~ A		*
B	*	
☞ C		

(2.24)

Here, stepping to B involves incurring an extra violation of constraint  $C1$ , both from A and C. Candidate C being better than A does not matter for they are not neighbours—otherwise they would not be local optima. The situation is fully symmetric from the viewpoint of the probabilities, because escaping from

---

<sup>27</sup>Here, I am presenting a caricature of the situation, because escaping from state C always has some minor positive probability in a real-valued simulated annealing. The train of thought should nevertheless be clear.

C has always the same chance as escaping from A: the violation profile difference incurs one violation of C1 in both cases. If you succeed to climb from A to B, you will also succeed to climb from C to B. If you are caught in C, you are also caught in A. The difference between A and C is invisible, and the trick that worked before does not work now. The distribution of the outputs will only depend on how candidate B distributes the probabilities between A and C—independently of the cooling schedule.

Can we make the chance of moving from A to B higher than that of moving from C to B in this last situation, too (at least in some phase of the simulation)? Remember that the transition probability  $P(A \rightarrow B|T)$  cannot depend on the violation profile of candidate C.

Changing the *a priori* probabilities of moving from B to A or C results in a distribution different from 50%-50%, because the chance of going to A or to C from B is not symmetric anymore; and yet, this distribution is still independent of the cooling schedule. Indeed, the *a priori* probabilities define the horizontal structure of the landscape, whereas our problem here concerns its vertical structure, to which the cooling schedule is related, as well.

This question is still an open research question, and may lead to altering the whole concept of simulated annealing for OT. Meanwhile, I can only imagine solutions that would not only contradict the general philosophy of OT, but also introduce further problems.

For instance, take the following new definition of the difference of two violation profiles. Start with the mark cancellation procedure from the highest ranked constraint. Initial violation marks shared by both candidates do not interest us. Suppose C1 is the fatal constraint where the better candidate (candidate A) has fewer violation marks than candidate B. Until here, the difference of the two violation profiles was defined as the constraint C1 and the number of uncanceled marks by C1. Now, we also want to take into consideration the highest violation mark of candidate A that is below C1. Eyeballing tableau (2.24), we can see that this is how one may capture the fact that moving from C to B involves a bigger upward step than from A to B.

Nonetheless, how to implement this idea in the case of the following tableau?

		C1	C2	C3
~	A		*	*
	B	*		
⊗	C		*	

(2.25)

Both A and C have their first violation mark at the same constraint C2, and their difference shows up only deeper in the hierarchy. Hence, this new definition would make no difference between moving from A or from C to B. Remember also that OT suggests not to look at further constraints if you have found the needed differences when comparing two candidates.

As a side remark, as SA-OT may predict the same probability to candidates A and C here, we have just shown that SA-OT does not have ganging-up cumulativity (cf. tableau (1.16) in section 1.3.5). Namely, the probability of candidate A (and similarly to C) does not change from tableau (2.25) to the following tableau, notwithstanding its different behaviour for constraint C3:

	C1	C2	C3
☞ A		*	
B	*		
~ C		*	*

(2.26)

Further, the new definition proposed would work incorrectly in this case:

	C1	C2	C3	C4
~ A	*			*
B	*	*		*
☞ C		*		*

(2.27)

Now, when comparing A to B, we have to descend two constraints in order to find the first violation mark incurred by A below the critical constraint C2. Two levels is a higher difference than the one level needed when we compare C to B. Obviously, we may consider again the highest constraint where the two profiles differ, and then the given differences can be said to be “two levels from C2” as opposed to “one level from C1”.

In addition, the number of violation marks found at a lower level by the better candidate needs also to be taken into consideration, as shown by the following tableau:

	C1	C2
~ A		**
B	*	
☞ C		*

(2.28)

We may speculate further. Nevertheless, I have no idea how to capture all these observations into a single elegant model. Do not forget that the only information at hand when determining the transition probabilities are the two violation profiles, and we cannot refer to other neighbours of the target state. Probably, the phenomenon discussed in the present subsection is an inevitable consequence of applying simulated annealing to a non-real valued function, such as is the case for the harmony function in Optimality Theory.<sup>28</sup>

One may also add a small bit of memory to the random-walker: while wandering around in the search space, the best candidate found so far is always remembered and compared to the present position. Then, the algorithm returns not necessarily the final position, but the best candidate found during the walk. This trick would work, especially in small search spaces as those seen in this section, but not necessarily in large ones. Nonetheless, we shall leave

---

<sup>28</sup>As proposed by Balázs Szendrői, a solution may be the approximation of the harmony function of OT with polynomials  $P(w)[q] = C_N(w)q^N + \dots + C_1(w)q + C_0(w)$  the coefficients of which are the constraint violation levels (using exponential weights; cf. subsection 3.3, and e.g. Smolensky’s pre-OT *Harmony Theory* or Prince (2002)). By increasing  $q$ , we may better approach the OT harmony function, but an unreasonably high value for  $q$  will simply reproduce the described situations. An additional argument for keeping  $q$  relatively low is that if the neighbouring candidates do not differ much, their violation profile is also similar, consequently smaller  $qs$  will also correctly account for their difference in harmony. Yet, one can also reproduce cumulativity effects (cf. subsection 1.3.5) this way—supposing one wishes to. As higher  $qs$  require higher ranges for the temperature, further research may also consider the possibility of gradually increasing  $q$  instead of decreasing the temperature.

this possibility for future work, and rather focus on what the implementation of traditional simulated annealing to OT can propose us.

Additionally, we shall later turn these failures of SA-OT into an advantage by claiming that such failures correspond to empirically attested agrammaticalities. Language—even slow and careful speech—can display irregular forms that contradict the general rules of the grammar, and yet, they are inevitably produced in speech and attested in corpora. Instead of turning the grammar much more complex so that it account for these irregularities, we shall argue to keep the grammar model (the underlying OT-system) simple, and explain these forms on the language production level.

The OT-grammars discussed in this section have been very simple, as they included only a few candidates and a few constraints. One may ask then how the precision is influenced by the number of candidates and the number of constraints, but no simple answer can be given. A major disadvantage of SA-OT is that the interactions between the neighbourhood structure, the constraint hierarchy and the cooling schedule is so complex that it is often impossible to find out the behaviour of the system without running the simulations. The second part of my dissertation introduces different models, involving larger but finite, as well as infinite candidate sets, and displaying very different behaviours. If there are only few local optima, then the system's precision may be similar to the precision of those analysed so far; if, however, the huge candidate set is full of local optima, as will be the case with the different models of Chapter 7, then precision can drop drastically.

Before that, let us turn to a few theoretical, formal and mathematical issues related to Optimality Theory in general, and SA-OT in particular.

