合肥學院
HEFEI UNIVERSITY

Metaheuristics for Smart Manufacturing

6. Simulated Annealing

Thomas Weise · 汤卫思

tweise@hfuu.edu.cn · http://iao.hfuu.edu.cn

Hefei University, South Campus 2
Faculty of Computer Science and Technology
Institute of Applied Optimization
230601 Shushan District, Hefei, Anhui, China
Econ. & Tech. Devel. Zone, Jinxiu Dadao 99

合肥学院 南艳湖校区/南2区
计算机科学与技术系
应用优化研究所
中国 安徽省 合肥市 蜀山区 230601
经济技术开发区 锦绣大道99号

Outline

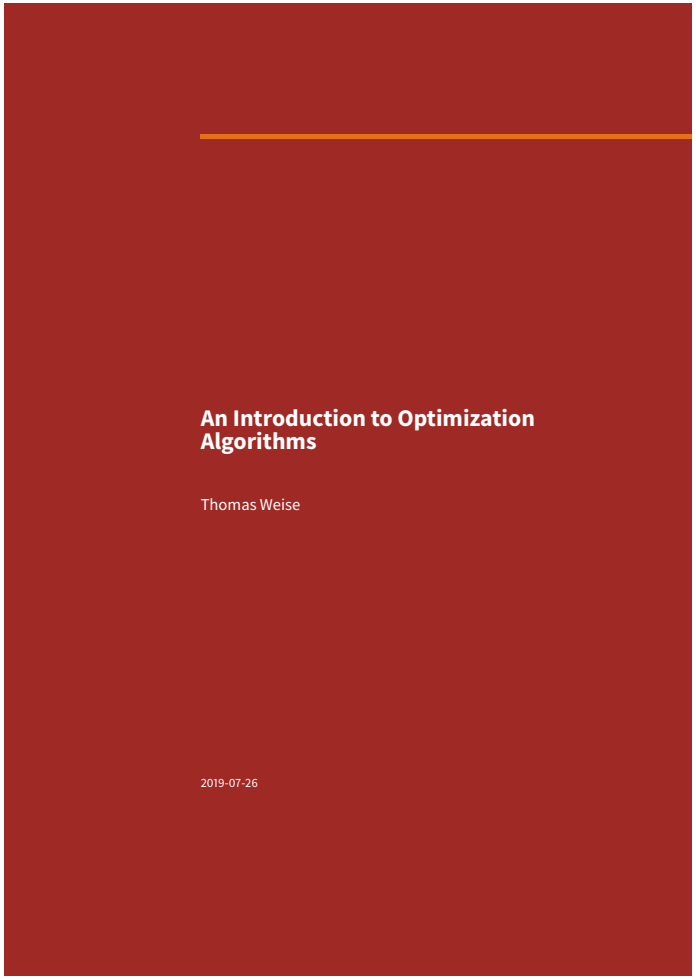
The slides are available at http://iao.hfuu.edu.cn/155, the book at http://thomasweise.github.io/aitoa, and the source code at http://www.github.com/thomasWeise/aitoa-code

course book

course material

An Introduction to Optimization Algorithms

The contents of this course are available as free electronic book *"An Introduction to Optimization Algorithms"* [1] at http://thomasweise.github.io/aitoa in pdf, html, azw3, and epub format, created with our bookbuildeR tool chain.



Section Outline

Idea

- So far, we have only discussed one variant of local search: the hill climbing algorithm.

Idea

- So far, we have only discussed one variant of local search: the hill climbing algorithm.
- A hill climbing algorithm is likely to get stuck at local optima, which may vary in quality.

Idea

- So far, we have only discussed one variant of local search: the hill climbing algorithm.
- A hill climbing algorithm is likely to get stuck at local optima, which may vary in quality.
- We found that we can utilize this variance of the result quality by restarting the optimization process when it could not improve any more.

Idea

- So far, we have only discussed one variant of local search: the hill climbing algorithm.
- A hill climbing algorithm is likely to get stuck at local optima, which may vary in quality.
- We found that we can utilize this variance of the result quality by restarting the optimization process when it could not improve any more.
- Such a restart is costly, as it forces the local search to start completely from scratch (while we, of course, remember the best-ever solution in a variable hidden from the algorithm).

Idea

- A schedule which is a local optimum probably is somewhat similar to what the globally optimal schedule would look like.

Idea

- A schedule which is a local optimum probably is somewhat similar to what the globally optimal schedule would look like.
- It must, obviously, also be somewhat different.

Idea

- A schedule which is a local optimum probably is somewhat similar to what the globally optimal schedule would look like.
- It must, obviously, also be somewhat different.
- This difference is shaped such that it cannot be conquered by the unary search operator that we use.

Idea

- A schedule which is a local optimum probably is somewhat similar to what the globally optimal schedule would look like.
- It must, obviously, also be somewhat different.
- This difference is shaped such that it cannot be conquered by the unary search operator that we use.
- If we do a restart, we also dispose of the similarities to the global optimum that we have already discovered.

Idea

- A schedule which is a local optimum probably is somewhat similar to what the globally optimal schedule would look like.
- It must, obviously, also be somewhat different.
- This difference is shaped such that it cannot be conquered by the unary search operator that we use.
- If we do a restart, we also dispose of the similarities to the global optimum that we have already discovered.
- We will subsequently spend time to re-discover them in the hope that this will happen in a way that allows us to eventually reach the global optimum itself.

Idea

- A schedule which is a local optimum probably is somewhat similar to what the globally optimal schedule would look like.
- It must, obviously, also be somewhat different.
- This difference is shaped such that it cannot be conquered by the unary search operator that we use.
- If we do a restart, we also dispose of the similarities to the global optimum that we have already discovered.
- We will subsequently spend time to re-discover them in the hope that this will happen in a way that allows us to eventually reach the global optimum itself.
- But maybe there is a less-costly way?

Simulated Annealing

- Simulated Annealing (SA) [2–5] is a local search which provides another approach to escape local optima [6, 7].

Simulated Annealing

- Simulated Annealing (SA) [2–5] is a local search which provides another approach to escape local optima [6, 7].
- Instead of restarting the algorithm when reaching a local optima, it tries to preserve the parts of the current best solution by permitting search steps towards worsening objective values.

Simulated Annealing

- Simulated Annealing (SA) [2–5] is a local search which provides another approach to escape local optima [6, 7].
- Instead of restarting the algorithm when reaching a local optima, it tries to preserve the parts of the current best solution by permitting search steps towards worsening objective values.
- This algorithm therefore introduces three principles

Simulated Annealing

- Simulated Annealing (SA) [2–5] is a local search which provides another approach to escape local optima [6, 7].
- Instead of restarting the algorithm when reaching a local optima, it tries to preserve the parts of the current best solution by permitting search steps towards worsening objective values.
- This algorithm therefore introduces three principles:
  1. Worse candidate solutions are sometimes accepted, too.

Simulated Annealing

- Simulated Annealing (SA) [2–5] is a local search which provides another approach to escape local optima [6, 7].
- Instead of restarting the algorithm when reaching a local optima, it tries to preserve the parts of the current best solution by permitting search steps towards worsening objective values.
- This algorithm therefore introduces three principles:
  1. Worse candidate solutions are sometimes accepted, too.
  2. The probability P of accepting them is decreases with increasing differences ΔE of the objective values to the current best solution.

Simulated Annealing

- Simulated Annealing (SA) [2–5] is a local search which provides another approach to escape local optima [6, 7].
- Instead of restarting the algorithm when reaching a local optima, it tries to preserve the parts of the current best solution by permitting search steps towards worsening objective values.
- This algorithm therefore introduces three principles:
  1. Worse candidate solutions are sometimes accepted, too.
  2. The probability P of accepting them is decreases with increasing differences ΔE of the objective values to the current best solution.
  3. The probability also decreases with the number of performed search steps.

Section Outline

Introduction: Annealing

Annealing of Metal

- Cold working metal causes/increases defects in crystal structure

Annealing of Metal

- Cold working metal causes/increases defects in crystal structure
- After cold working, annealing [8] is performed

Annealing of Metal

- Cold working metal causes/increases defects in crystal structure
- After cold working, annealing [8] is performed
- The metal is heated to, like 0.4 * melting temperature

Annealing of Metal

- Cold working metal causes/increases defects in crystal structure
- After cold working, annealing [8] is performed
- The metal is heated to, like 0.4 * melting temperature
- Ions inside metal can move around

Annealing of Metal

- Cold working metal causes/increases defects in crystal structure
- After cold working, annealing [8] is performed
- The metal is heated to, like 0.4 * melting temperature
- Ions inside metal can move around
- Metal is slowly cooled down, ions assume low-energy, stable positions in crystal $\rightarrow$ metal becomes more stable

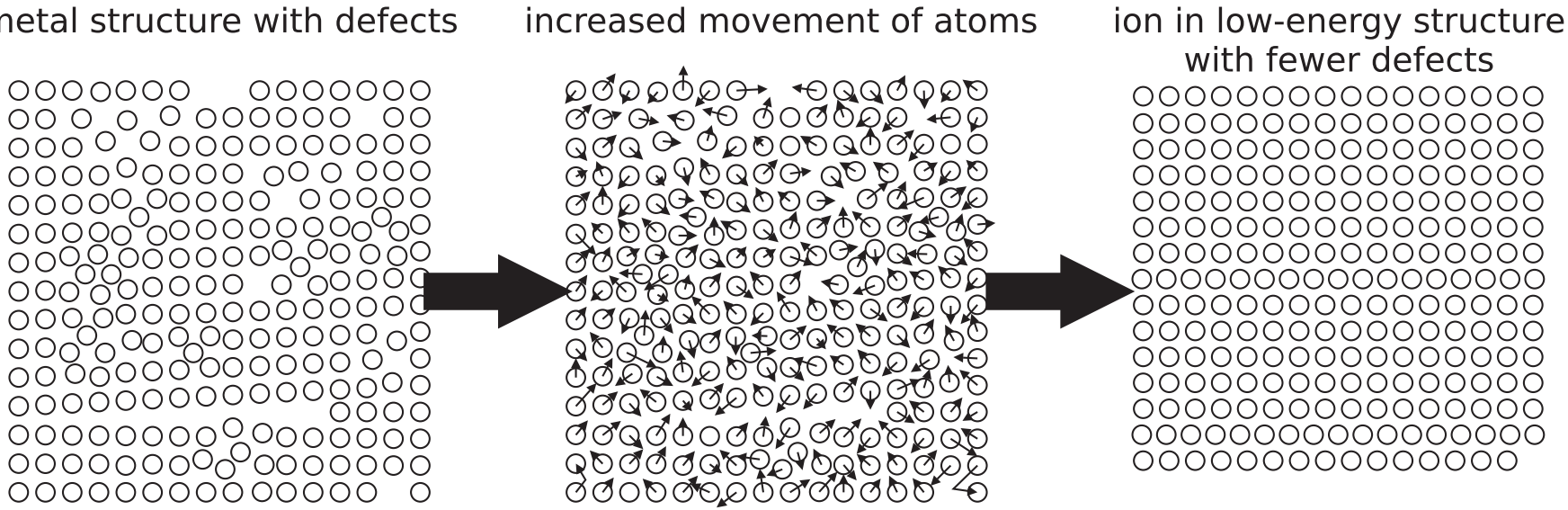
Annealing of Metal

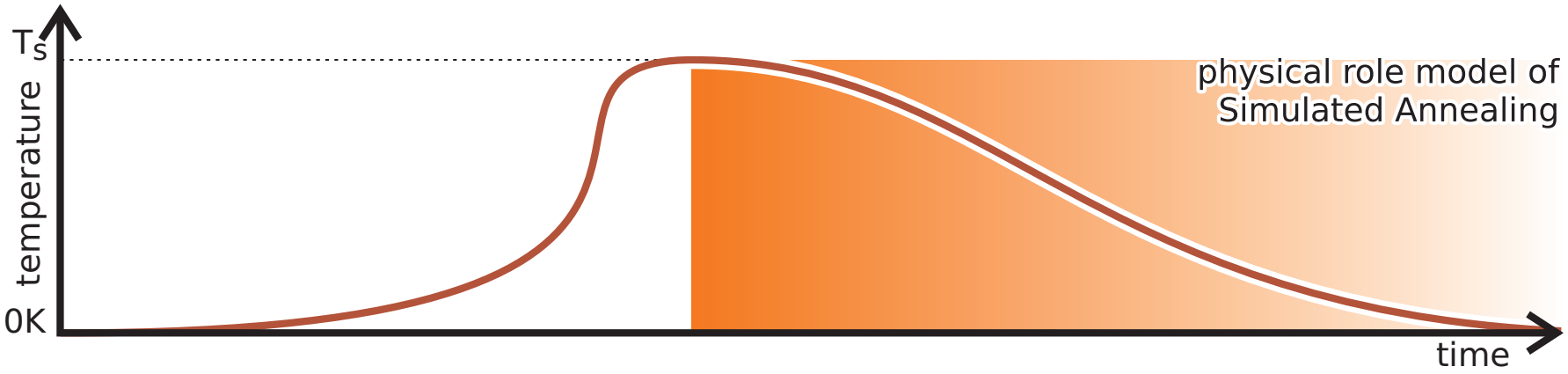
- Cold working metal causes/increases defects in crystal structure
- After cold working, annealing [8] is performed
- The metal is heated to, like 0.4 * melting temperature
- Ions inside metal can move around
- Metal is slowly cooled down, ions assume low-energy, stable positions in crystal $\rightarrow$ metal becomes more stable
- Due to their movement, ions may temporarily assume positions of high energy

Annealing of Metal

- Cold working metal causes/increases defects in crystal structure
- After cold working, annealing [8] is performed
- The metal is heated to, like 0.4 * melting temperature
- Ions inside metal can move around
- Metal is slowly cooled down, ions assume low-energy, stable positions in crystal $\rightarrow$ metal becomes more stable
- Due to their movement, ions may temporarily assume positions of high energy
- An initial, brittle crystal structure is transformed to a much better configuration by stepping over good and bad states

Introduction: Annealing





Metropolis Algorithm

- Metropolis [9] wants to simulate this process.

Metropolis Algorithm

- Metropolis [9] wants to simulate this process.
- First, we need to understand: What is temperature T?

Metropolis Algorithm

- Metropolis [9] wants to simulate this process.
- First, we need to understand: What is temperature T?
- Each material consists of many different particles (atoms, ions, molecules, etc.)

Metropolis Algorithm

- Metropolis [9] wants to simulate this process.
- First, we need to understand: What is temperature T?
- Each material consists of many different particles
- The micro-state of a material is the tuple of the positions and velocities of all particles – this is uninteresting

Metropolis Algorithm

- Metropolis [9] wants to simulate this process.
- First, we need to understand: What is temperature T?
- Each material consists of many different particles
- The micro-state of a material is the tuple of the positions and velocities of all particles – this is uninteresting
- With each such state, there is an energy E associated

Metropolis Algorithm

- Metropolis [9] wants to simulate this process.
- First, we need to understand: What is temperature T?
- Each material consists of many different particles
- The micro-state of a material is the tuple of the positions and velocities of all particles – this is uninteresting
- With each such state, there is an energy E associated: If many of the particles move around quickly or are in an unfavourable position, the energy E is high and if they form a nice crystal and don't move, the energy is low
- Now we consider a value of E as a macro-state of the system

Metropolis Algorithm

- Metropolis [9] wants to simulate this process.
- First, we need to understand: What is temperature T?
- Each material consists of many different particles
- The micro-state of a material is the tuple of the positions and velocities of all particles – this is uninteresting
- With each such state, there is an energy E associated: If many of the particles move around quickly or are in an unfavourable position, the energy E is high
- Now we consider a value of E as a macro-state of the system and to each such state, there belong many possible micro-states
- A system at temperature T has the probability $e^{-\frac{E}{k_B * T}}$ to be in a macro state with energy E.

Metropolis Algorithm

- Metropolis [9] wants to simulate this process.
- First, we need to understand: What is temperature T?
- Each material consists of many different particles
- The micro-state of a material is the tuple of the positions and velocities of all particles – this is uninteresting
- With each such state, there is an energy E associated: If many of the particles move around quickly or are in an unfavourable position, the energy E is high
- Now we consider a value of E as a macro-state of the system and to each such state, there belong many possible micro-states
- A system at temperature T has the probability $e^{-\frac{E}{k_B * T}}$ to be in a macro state with energy E.

$$k_b = 1.380650524 * 10^{-27} J/K \text{ is the Boltzmann constant} \quad (1)$$

Metropolis Algorithm

- Metropolis [9] wants to simulate this process.
- First, we need to understand: What is temperature T?
- Each material consists of many different particles
- The micro-state of a material is the tuple of the positions and velocities of all particles – this is uninteresting
- With each such state, there is an energy E associated: If many of the particles move around quickly or are in an unfavourable position, the energy E is high
- Now we consider a value of E as a macro-state of the system and to each such state, there belong many possible micro-states
- A system at temperature T has the probability $e^{-\frac{E}{k_B * T}}$ to be in a macro state with energy E.
- In other words: The higher the temperature, the higher the chance to be in a high-energy state

Metropolis Algorithm

- Based on this, Metropolis [9] develops a simulation for annealing in form of a randomized algorithm

Metropolis Algorithm

- Based on this, Metropolis [9] develops a simulation for annealing in form of a randomized algorithm
- s_1 be the current configuration of the ions and s_2 a possible new configuration, T be the temperature (decreasing over time)

Metropolis Algorithm

- Based on this, Metropolis [9] develops a simulation for annealing in form of a randomized algorithm
- s_1 be the current configuration of the ions and s_2 a possible new configuration, T be the temperature (decreasing over time)

$$\Delta E = E(s_2) - E(s_1) \tag{1}$$

Metropolis Algorithm

- Based on this, Metropolis [9] develops a simulation for annealing in form of a randomized algorithm
- s_1 be the current configuration of the ions and s_2 a possible new configuration, T be the temperature (decreasing over time)

$$\Delta E = E(s_2) - E(s_1) \tag{1}$$

- ΔE is the energy difference between the states

Metropolis Algorithm

- Based on this, Metropolis [9] develops a simulation for annealing in form of a randomized algorithm
- s_1 be the current configuration of the ions and s_2 a possible new configuration, T be the temperature (decreasing over time)

$$\Delta E = E(s_2) - E(s_1) \tag{1}$$

- ΔE is the energy difference between the states

$$P(\Delta E) = \begin{cases} e^{-\frac{\Delta E}{k_B * T}} & \text{if } \Delta E > 0 \\ 1 & \text{otherwise} \end{cases} \tag{2}$$

Metropolis Algorithm

- Based on this, Metropolis [9] develops a simulation for annealing in form of a randomized algorithm
- s_1 be the current configuration of the ions and s_2 a possible new configuration, T be the temperature (decreasing over time)

$$\Delta E = E(s_2) - E(s_1) \tag{1}$$

- ΔE is the energy difference between the states

$$P(\Delta E) = \begin{cases} e^{-\frac{\Delta E}{k_B * T}} & \text{if } \Delta E > 0 \\ 1 & \text{otherwise} \end{cases} \tag{2}$$

- $P(\Delta E)$ is the probability that the new state s_2 will be accepted

From Simulating Annealing to Simulated Annealing

- Metropolis' simulation has the following behavior

From Simulating Annealing to Simulated Annealing

- Metropolis' simulation has the following behavior:
  1. It starts with a random, probably bad state.

From Simulating Annealing to Simulated Annealing

- Metropolis' simulation has the following behavior:
  1. It starts with a random, probably bad state.
  2. In each step, a new state which is similar to the current state is created.

From Simulating Annealing to Simulated Annealing

- Metropolis' simulation has the following behavior:
  1. It starts with a random, probably bad state.
  2. In each step, a new state which is similar to the current state is created.
  3. If the new state is better, it becomes the current state.

From Simulating Annealing to Simulated Annealing

- Metropolis' simulation has the following behavior:
  1. It starts with a random, probably bad state.
  2. In each step, a new state which is similar to the current state is created.
  3. If the new state is better, it becomes the current state.
  4. If it is worse, it may stil be accepted as current state with a certain probability.

From Simulating Annealing to Simulated Annealing

- Metropolis' simulation has the following behavior:
  1. It starts with a random, probably bad state.
  2. In each step, a new state which is similar to the current state is created.
  3. If the new state is better, it becomes the current state.
  4. If it is worse, it may stil be accepted as current state with a certain probability.
  5. In the end, the system usually arrives in a very good state.

From Simulating Annealing to Simulated Annealing

- Metropolis' simulation has the following behavior:
  1. It starts with a random, probably bad state.
  2. In each step, a new state which is similar to the current state is created.
  3. If the new state is better, it becomes the current state.
  4. If it is worse, it may stil be accepted as current state with a certain probability.
  5. In the end, the system usually arrives in a very good state.
- Wait.

From Simulating Annealing to Simulated Annealing

- Metropolis' simulation has the following behavior:
  1. It starts with a random, probably bad solution.
  2. In each step, a new solution which is similar to the current solution is created using the unary search operation.
  3. If the new state is better, it becomes the current solution.
  4. If it is worse, it may stil be accepted as current solution with a certain probability.
  5. In the end, the system usually arrives inat a very good solution.
- Wait. Could we just...

From Simulating Annealing to Simulated Annealing

- Metropolis' simulation has the following behavior:
  1. It starts with a random, probably bad solution.
  2. In each step, a new solution which is similar to the current solution is created using the unary search operation.
  3. If the new state is better, it becomes the current solution.
  4. If it is worse, it may stil be accepted as current solution with a certain probability.
  5. In the end, the system usually arrives at a very good solution.
- Wait. Could we just... ...and so Simulated Annealing was invented.

Section Outline

Realization of Idea

- ΔE be the difference between the objective value of the freshly sampled point x' from the search space and the current best point x

Realization of Idea

- ΔE be the difference between the objective value of the freshly sampled point x' from the search space and the current best point x:

$$\Delta E = f(\gamma(x')) - f(\gamma(x)) \tag{3}$$

Realization of Idea

- ΔE be the difference between the objective value of the freshly sampled point x' from the search space and the current best point x:

$$\Delta E = f(\gamma(x')) - f(\gamma(x)) \tag{3}$$

  - $\Delta E < 0$ means that x' is better than x since $f(\gamma(x')) < f(\gamma(x))$.

Realization of Idea

- ΔE be the difference between the objective value of the freshly sampled point x' from the search space and the current best point x:

$$\Delta E = f(\gamma(x')) - f(\gamma(x)) \tag{3}$$

  - $\Delta E < 0$ means that x' is better than x since $f(\gamma(x')) < f(\gamma(x))$.
  - $\Delta E > 0$ means that the new solution is worse.

Realization of Idea

- ΔE be the difference between the objective value of the freshly sampled point x' from the search space and the current best point x:

$$\Delta E = f(\gamma(x')) - f(\gamma(x)) \tag{3}$$

- The probability P to overwrite x with x' then be

$$P = \begin{cases} 1 & \text{if } \Delta E \leq 0 \\ e^{-\frac{\Delta E}{T}} & \text{if } \Delta E > 0 \wedge T > 0 \\ 0 & \text{otherwise } (\Delta E > 0 \wedge T = 0) \end{cases} \tag{4}$$

Realization of Idea

- ΔE be the difference between the objective value of the freshly sampled point x' from the search space and the current best point x:

$$\Delta E = f(\gamma(x')) - f(\gamma(x)) \tag{3}$$

- The probability P to overwrite x with x' then be

$$P = \begin{cases} 1 & \text{if } \Delta E \leq 0 \\ e^{-\frac{\Delta E}{T}} & \text{if } \Delta E > 0 \wedge T > 0 \\ 0 & \text{otherwise } (\Delta E > 0 \wedge T = 0) \end{cases} \tag{4}$$

  - If the new candidate solution is actually better than the current best one, i.e., $\Delta E < 0$, then we will definitely accept it.

Realization of Idea

- ΔE be the difference between the objective value of the freshly sampled point x' from the search space and the current best point x:

$$\Delta E = f(\gamma(x')) - f(\gamma(x)) \tag{3}$$

- The probability P to overwrite x with x' then be

$$P = \begin{cases} 1 & \text{if } \Delta E \leq 0 \\ e^{-\frac{\Delta E}{T}} & \text{if } \Delta E > 0 \wedge T > 0 \\ 0 & \text{otherwise } (\Delta E > 0 \wedge T = 0) \end{cases} \tag{4}$$

  - If the new candidate solution is actually better than the current best one, i.e., $\Delta E < 0$, then we will definitely accept it.
  - If the new solution is worse ($\Delta E > 0$), the acceptance probability then

Realization of Idea

- ΔE be the difference between the objective value of the freshly sampled point x' from the search space and the current best point x:

$$\Delta E = f(\gamma(x')) - f(\gamma(x)) \tag{3}$$

- The probability P to overwrite x with x' then be

$$P = \begin{cases} 1 & \text{if } \Delta E \leq 0 \\ e^{-\frac{\Delta E}{T}} & \text{if } \Delta E > 0 \wedge T > 0 \\ 0 & \text{otherwise } (\Delta E > 0 \wedge T = 0) \end{cases} \tag{4}$$

  - If the new candidate solution is actually better than the current best one, i.e., $\Delta E < 0$, then we will definitely accept it.
  - If the new solution is worse ($\Delta E > 0$), the acceptance probability then gets smaller the larger ΔE is

Realization of Idea

- ΔE be the difference between the objective value of the freshly sampled point x' from the search space and the current best point x:

$$\Delta E = f(\gamma(x')) - f(\gamma(x)) \tag{3}$$

- The probability P to overwrite x with x' then be

$$P = \begin{cases} 1 & \text{if } \Delta E \leq 0 \\ e^{-\frac{\Delta E}{T}} & \text{if } \Delta E > 0 \wedge T > 0 \\ 0 & \text{otherwise } (\Delta E > 0 \wedge T = 0) \end{cases} \tag{4}$$

  - If the new candidate solution is actually better than the current best one, i.e., $\Delta E < 0$, then we will definitely accept it.
  - If the new solution is worse ($\Delta E > 0$), the acceptance probability then gets smaller the larger ΔE is and gets smaller the smaller the so-called "temperature" $T \geq 0$ is.

Annealing

- The temperature $T > 0$ and the objective value difference $\Delta E > 0$ enter the equation in an exponential term $e^{-\frac{\Delta E}{T}}$ and it holds that $e^{-a} < e^{-b} \forall a > b$ and $e^{-a} \in [0, 1] \forall a > 0$.

Annealing

- The temperature $T > 0$ and the objective value difference $\Delta E > 0$ enter the equation in an exponential term $e^{-\frac{\Delta E}{T}}$ and it holds that $e^{-a} < e^{-b} \forall a > b$ and $e^{-a} \in [0, 1] \forall a > 0$.
- The temperature decreases and approaches zero with the algorithm iteration τ, i.e., the performed objective function evaluations.

Annealing

- The temperature $T > 0$ and the objective value difference $\Delta E > 0$ enter the equation in an exponential term $e^{-\frac{\Delta E}{T}}$ and it holds that $e^{-a} < e^{-b} \forall a > b$ and $e^{-a} \in [0, 1] \forall a > 0$.
- The temperature decreases and approaches zero with the algorithm iteration τ, i.e., the performed objective function evaluations.
- The optimization process is initially "hot."

Annealing

- The temperature $T > 0$ and the objective value difference $\Delta E > 0$ enter the equation in an exponential term $e^{-\frac{\Delta E}{T}}$ and it holds that $e^{-a} < e^{-b} \forall a > b$ and $e^{-a} \in [0, 1] \forall a > 0$.
- The temperature decreases and approaches zero with the algorithm iteration τ, i.e., the performed objective function evaluations.
- The optimization process is initially "hot."
- Then, the search progresses wildly any may accept even significantly worse solutions.

Annealing

- The temperature $T > 0$ and the objective value difference $\Delta E > 0$ enter the equation in an exponential term $e^{-\frac{\Delta E}{T}}$ and it holds that $e^{-a} < e^{-b} \forall a > b$ and $e^{-a} \in [0, 1] \forall a > 0$.
- The temperature decreases and approaches zero with the algorithm iteration τ, i.e., the performed objective function evaluations.
- The optimization process is initially "hot."
- Then, the search progresses wildly any may accept even significantly worse solutions.
- As the process "cools" down, the search tends to accept fewer and fewer worse solutions and more likely such which are only a bit worse.

Annealing

- The temperature $T > 0$ and the objective value difference $\Delta E > 0$ enter the equation in an exponential term $e^{-\frac{\Delta E}{T}}$ and it holds that $e^{-a} < e^{-b} \forall a > b$ and $e^{-a} \in [0, 1] \forall a > 0$.
- The temperature decreases and approaches zero with the algorithm iteration τ, i.e., the performed objective function evaluations.
- The optimization process is initially "hot."
- Then, the search progresses wildly any may accept even significantly worse solutions.
- As the process "cools" down, the search tends to accept fewer and fewer worse solutions and more likely such which are only a bit worse.
- Eventually, at temperature $T = 0$, the algorithm only accepts better solutions.

Annealing

- The temperature $T > 0$ and the objective value difference $\Delta E > 0$ enter the equation in an exponential term $e^{-\frac{\Delta E}{T}}$ and it holds that $e^{-a} < e^{-b} \forall a > b$ and $e^{-a} \in [0, 1] \forall a > 0$.
- The temperature decreases and approaches zero with the algorithm iteration τ, i.e., the performed objective function evaluations.
- The optimization process is initially "hot."
- Then, the search progresses wildly any may accept even significantly worse solutions.
- As the process "cools" down, the search tends to accept fewer and fewer worse solutions and more likely such which are only a bit worse.
- Eventually, at temperature $T = 0$, the algorithm only accepts better solutions.
- T is actually a monotonously decreasing function $T(\tau)$ called the "temperature schedule" and it holds that $\lim_{\tau \to \infty} T(\tau) = 0$.

Temperature Schedule

- There are two common types of temperature schedules, i.e., methods to decrease over time

Temperature Schedule

- There are two common types of temperature schedules, i.e., methods to decrease over time
  1. the exponential schedule, with start temperature T_s, it has a parameter $\epsilon \in (0, +\infty)$

Temperature Schedule

- There are two common types of temperature schedules, i.e., methods to decrease over time
  1. the exponential schedule, with start temperature T_s, it has a parameter $\epsilon \in (0, +\infty)$

$$T(\tau) = T_s * (1 - \epsilon)^{\tau - 1} \tag{5}$$

Temperature Schedule

- There are two common types of temperature schedules, i.e., methods to decrease over time
  1. the exponential schedule, with start temperature T_s, it has a parameter $\epsilon \in (0, +\infty)$

$$T(\tau) = T_s * (1 - \epsilon)^{\tau - 1} \tag{5}$$

  2. the logarithmic schedule, with start temperature T_s, it has a parameter $\epsilon \in (0, 1)$

Temperature Schedule

- There are two common types of temperature schedules, i.e., methods to decrease over time
  1. the exponential schedule, with start temperature T_s, it has a parameter $\epsilon \in (0, +\infty)$

$$T(\tau) = T_s * (1 - \epsilon)^{\tau - 1} \tag{5}$$

  2. the logarithmic schedule, with start temperature T_s, it has a parameter $\epsilon \in (0, 1)$

$$T(\tau) = \frac{T_s}{\ln\left(\epsilon(\tau - 1) + e\right)} \tag{6}$$

Abstract Class for Temperature Schedules

Listing: Abstract Class for Temperature Schedules.

```
public abstract class TemperatureSchedule {

  public abstract double temperature(long tau);

}
```

Exponential Temperature Schedule

Listing: Exponential Temperature Schedule.

```
public class Exponential extends TemperatureSchedule {

  public double temperature(long tau) {
    return (this.startTemperature * Math.pow(
          (1d - this.epsilon), (tau - 1L)));
  }
}
```

Logarithmic Temperature Schedule

Listing: Logarithmic Temperature Schedule.

```
public class Logarithmic extends TemperatureSchedule {

public double temperature(long tau) {
    if (tau >= Long.MAX_VALUE) return 0d;
    return (this.startTemperature / Math.log(
              ((tau - 1L) * this.epsilon) + Math.E));
  }
}
```

How to use?

- We want to use the temperature schedules such that the probability of accepting reasonably worse solution decreases reasonably smoothly during the optimization process.

How to use?

- We want to use the temperature schedules such that the probability of accepting reasonably worse solution decreases reasonably smoothly during the optimization process.
- We have 3min of runtime.

How to use?

- We want to use the temperature schedules such that the probability of accepting reasonably worse solution decreases reasonably smoothly during the optimization process.
- We have 3min of runtime.
- We found from our previous experiments that a hill climber can do about 30'000'000 steps on `swv15` and 97'000'000 on `abz7` within these 3min.

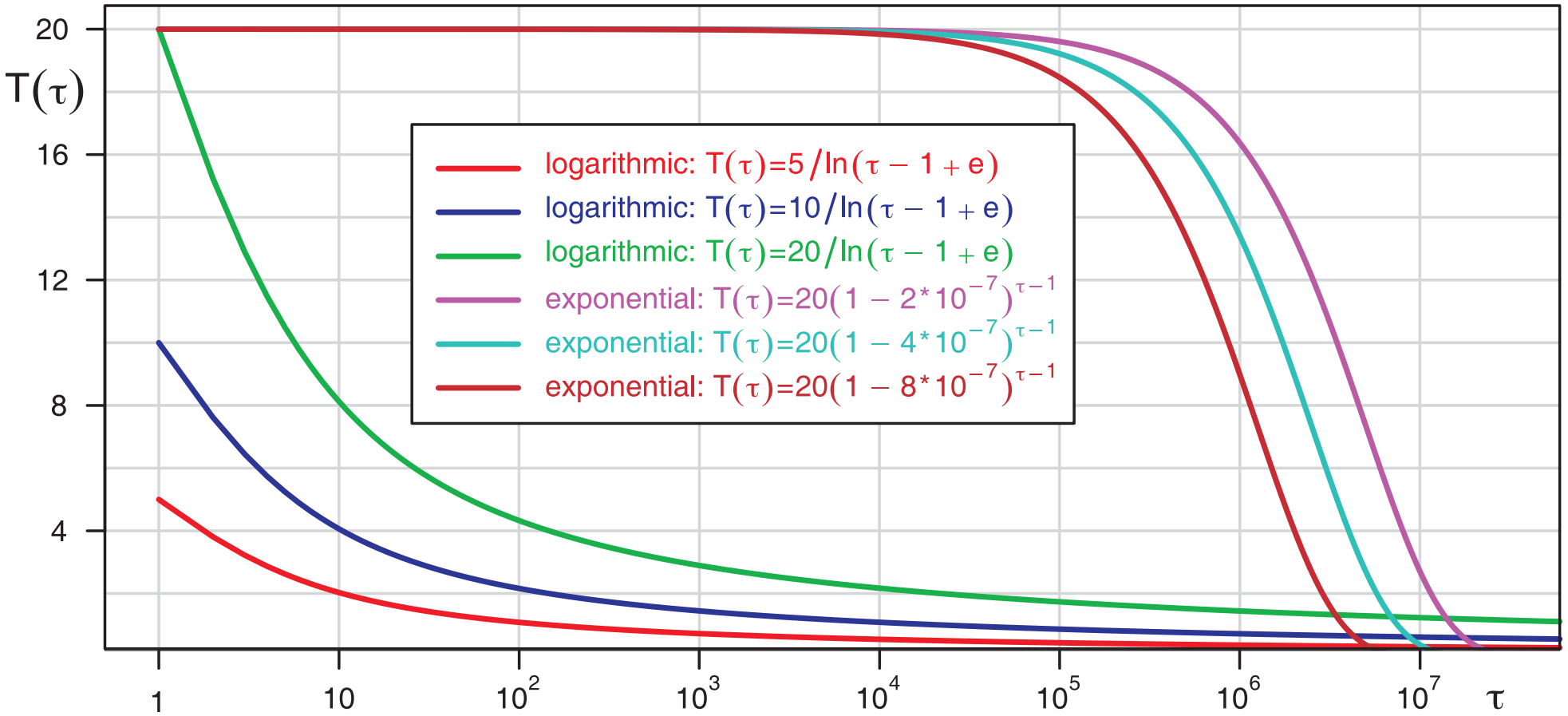
How to use?

- We want to use the temperature schedules such that the probability of accepting reasonably worse solution decreases reasonably smoothly during the optimization process.
- We have 3min of runtime.
- We found from our previous experiments that a hill climber can do about 30'000'000 steps on `swv15` and 97'000'000 on `abz7` within these 3min.
- We know that schedules that an acceptable scale of "worse" is somewhere around 1 to 10 from the best solutions we have seen so far.

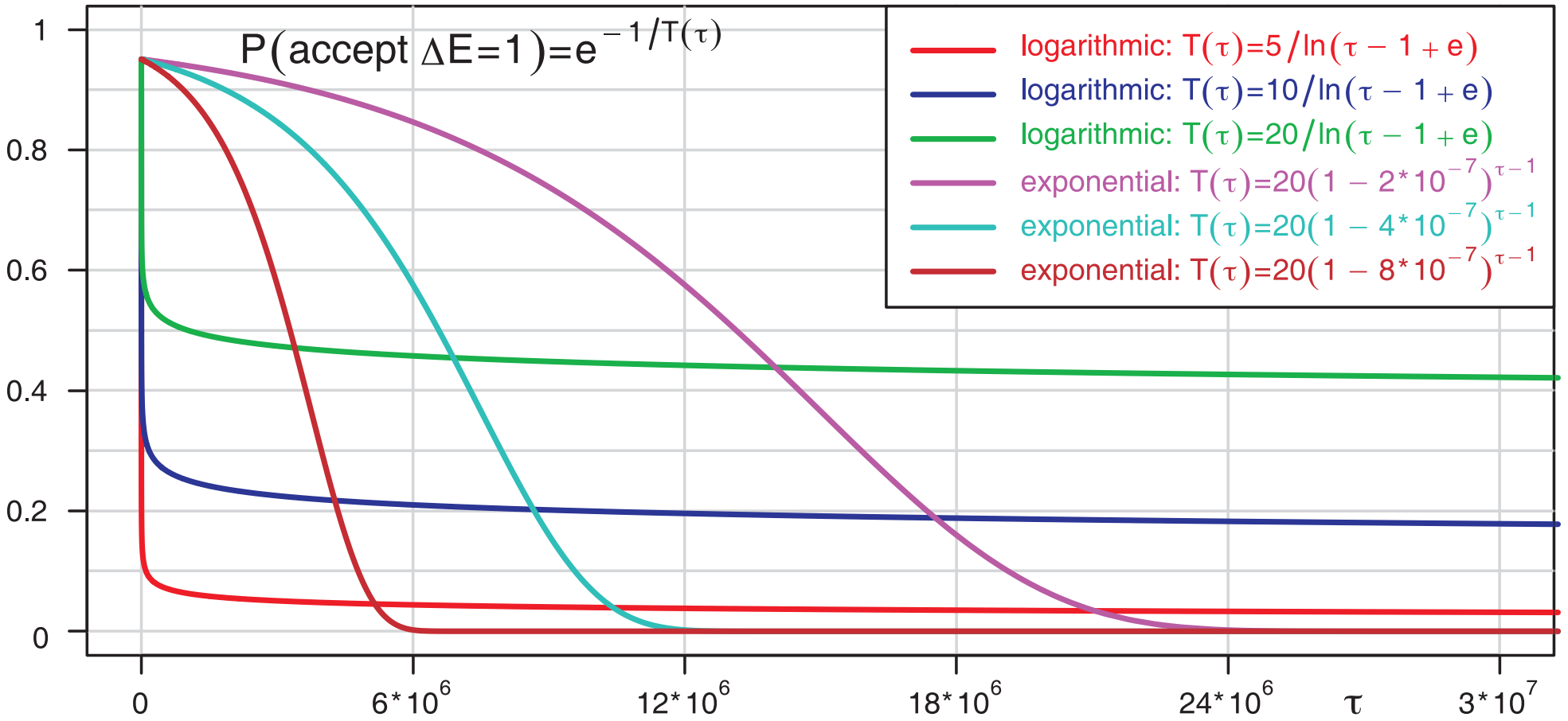
How to use?

- We want to use the temperature schedules such that the probability of accepting reasonably worse solution decreases reasonably smoothly during the optimization process.
- We have 3min of runtime.
- We found from our previous experiments that a hill climber can do about 30'000'000 steps on `swv15` and 97'000'000 on `abz7` within these 3min.
- We know that schedules that an acceptable scale of "worse" is somewhere around 1 to 10 from the best solutions we have seen so far.
- We make the following choices...

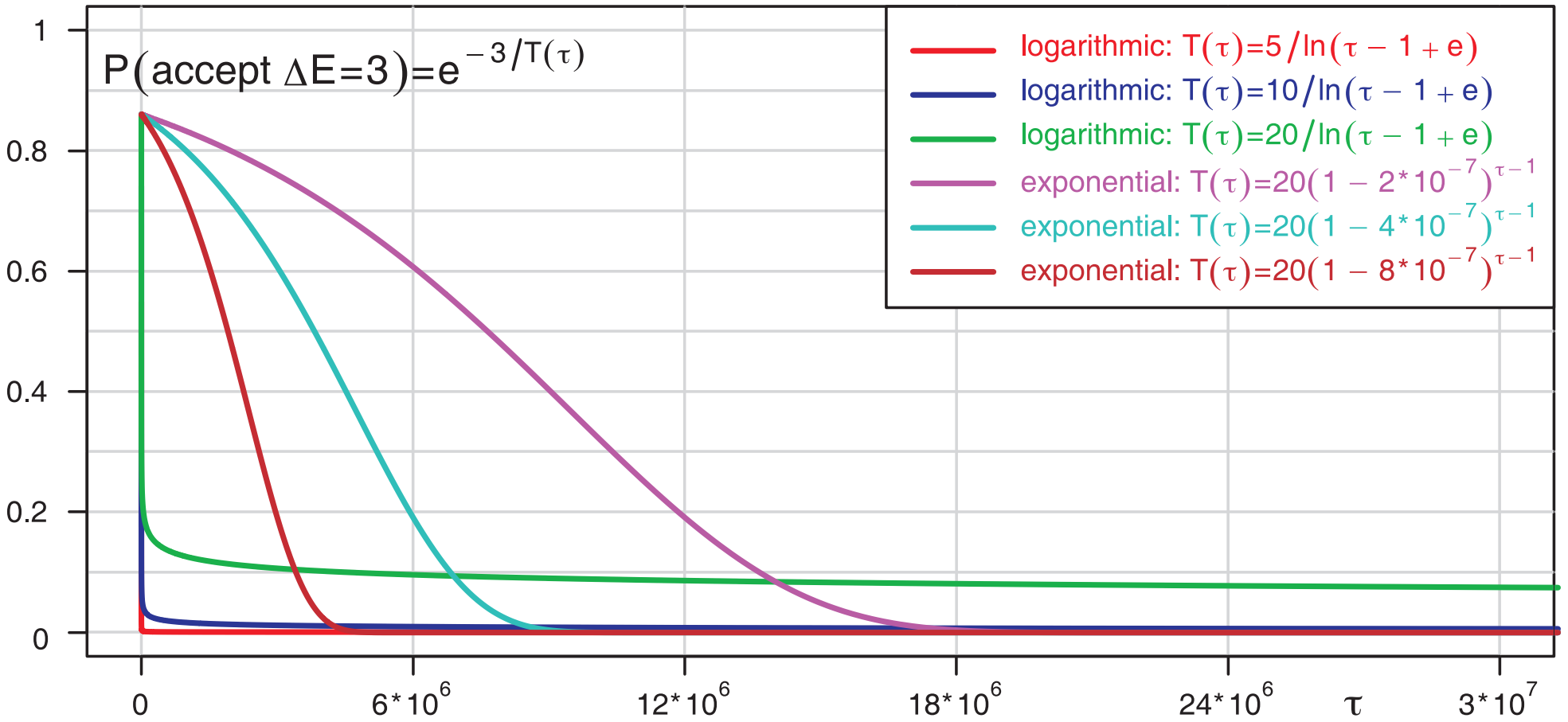
How to use?


20
16
12
8
4
T(τ)
1
10
10^2
10^3
10^4
10^5
10^6
10^7
τ
logarithmic: T(τ)=5/ln(τ − 1 + e)
logarithmic: T(τ)=10/ln(τ − 1 + e)
logarithmic: T(τ)=20/ln(τ − 1 + e)
exponential: T(τ)=20(1 − 2*10^−7)^τ−1
exponential: T(τ)=20(1 − 4*10^−7)^τ−1
exponential: T(τ)=20(1 − 8*10^−7)^τ−1

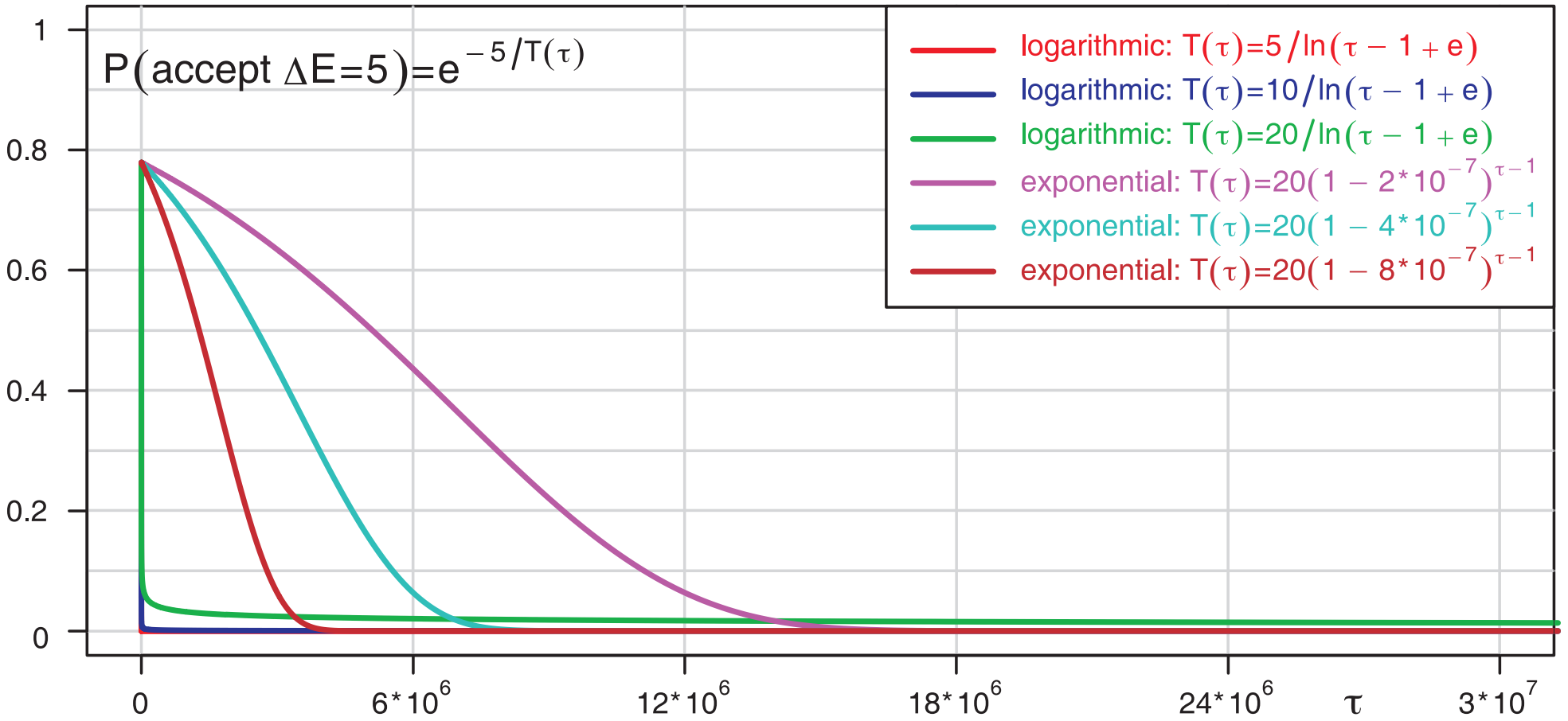
How to use?


P(accept ΔE=1)=e^{−1/T(τ)}
logarithmic: T(τ)=5/ln(τ − 1 + e)
logarithmic: T(τ)=10/ln(τ − 1 + e)
logarithmic: T(τ)=20/ln(τ − 1 + e)
exponential: T(τ)=20(1 − 2*10^{−7})^{τ−1}
exponential: T(τ)=20(1 − 4*10^{−7})^{τ−1}
exponential: T(τ)=20(1 − 8*10^{−7})^{τ−1}
1
0.8
0.6
0.4
0.2
0
0
6*10^6
12*10^6
18*10^6
24*10^6
τ
3*10^7

How to use?


P(accept ΔE=3)=e^{−3/T(τ)}
logarithmic: T(τ)=5/ln(τ − 1 + e)
logarithmic: T(τ)=10/ln(τ − 1 + e)
logarithmic: T(τ)=20/ln(τ − 1 + e)
exponential: T(τ)=20(1 − 2*10^{−7})^{τ−1}
exponential: T(τ)=20(1 − 4*10^{−7})^{τ−1}
exponential: T(τ)=20(1 − 8*10^{−7})^{τ−1}
1
0.8
0.6
0.4
0.2
0
0
6*10^6
12*10^6
18*10^6
24*10^6
τ
3*10^7

How to use?


P(accept ΔE=5)=e^{−5/T(τ)}
logarithmic: T(τ)=5/ln(τ − 1 + e)
logarithmic: T(τ)=10/ln(τ − 1 + e)
logarithmic: T(τ)=20/ln(τ − 1 + e)
exponential: T(τ)=20(1 − 2*10^{−7})^{τ−1}
exponential: T(τ)=20(1 − 4*10^{−7})^{τ−1}
exponential: T(τ)=20(1 − 8*10^{−7})^{τ−1}
1
0.8
0.6
0.4
0.2
0
0
6*10^6
12*10^6
18*10^6
24*10^6
τ
3*10^7

Simulated Annealing Algorithm

- Simulated Annealing = Hill Climbing + probabilistically accepting worse solutions

Simulated Annealing Algorithm

- Simulated Annealing = Hill Climbing + probabilistically accepting worse solutions
- Simple Concept

Simulated Annealing Algorithm

- Simulated Annealing = Hill Climbing + probabilistically accepting worse solutions
- Simple Concept:
  1. create random initial solution, which also becomes the first "current solution"

Simulated Annealing Algorithm

- Simulated Annealing = Hill Climbing + probabilistically accepting worse solutions
- Simple Concept:
  1. create random initial solution, which also becomes the first "current solution"
  2. make a modified copy of the current solution

Simulated Annealing Algorithm

- Simulated Annealing = Hill Climbing + probabilistically accepting worse solutions
- Simple Concept:
  1. create random initial solution, which also becomes the first "current solution"
  2. make a modified copy of the current solution
  3. if it is better: it becomes the new current solution

Simulated Annealing Algorithm

- Simulated Annealing = Hill Climbing + probabilistically accepting worse solutions
- Simple Concept:
  1. create random initial solution, which also becomes the first "current solution"
  2. make a modified copy of the current solution
  3. if it is better: it becomes the new current solution
  4. if it is worse: accept it as current solution with probability $P(\Delta E, \tau)$ (which gets smaller over time and also the smaller the worse the new solution is) or otherwise reject it.

Simulated Annealing Algorithm

- Simulated Annealing = Hill Climbing + probabilistically accepting worse solutions
- Simple Concept:
  1. create random initial solution, which also becomes the first "current solution"
  2. make a modified copy of the current solution
  3. if it is better: it becomes the new current solution
  4. if it is worse: accept it as current solution with probability $P(\Delta E, \tau)$ (which gets smaller over time and also the smaller the worse the new solution is) or otherwise reject it.
  5. go back to 2 (until the time is up)

Section Outline

So what do we get?

- We test the following configurations

So what do we get?

- We test the following configurations
  1. `sa_e_20_2e-7_1swap`: Simulated Annealing with Exponential Schedule at $T_s = 20$ with $\epsilon = 2 * 10^{-7}$

So what do we get?

- We test the following configurations
  1. `sa_e_20_2e-7_1swap`: Simulated Annealing with Exponential Schedule at $T_s = 20$ with $\epsilon = 2 * 10^{-7}$
  2. `sa_e_20_4e-7_1swap`: Simulated Annealing with Exponential Schedule at $T_s = 20$ with $\epsilon = 4 * 10^{-7}$

So what do we get?

- We test the following configurations
  1. `sa_e_20_2e-7_1swap`: Simulated Annealing with Exponential Schedule at $T_s = 20$ with $\epsilon = 2 * 10^{-7}$
  2. `sa_e_20_4e-7_1swap`: Simulated Annealing with Exponential Schedule at $T_s = 20$ with $\epsilon = 4 * 10^{-7}$
  3. `sa_e_20_8e-7_1swap`: Simulated Annealing with Exponential Schedule at $T_s = 20$ with $\epsilon = 8 * 10^{-7}$

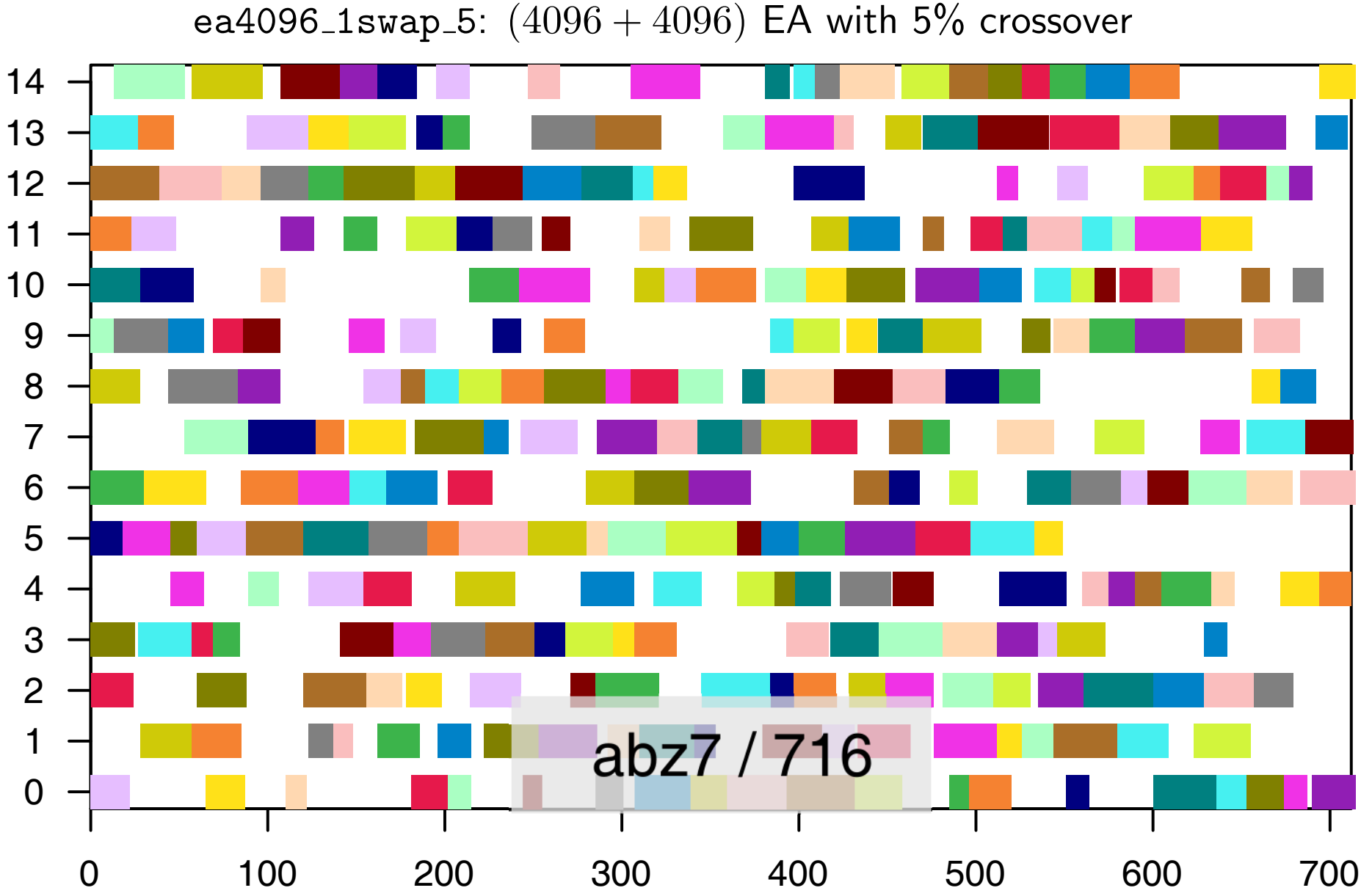
So what do we get?

- We test the following configurations
  1. `sa_e_20_2e-7_1swap`: Simulated Annealing with Exponential Schedule at $T_s = 20$ with $\epsilon = 2 * 10^{-7}$
  2. `sa_e_20_4e-7_1swap`: Simulated Annealing with Exponential Schedule at $T_s = 20$ with $\epsilon = 4 * 10^{-7}$
  3. `sa_e_20_8e-7_1swap`: Simulated Annealing with Exponential Schedule at $T_s = 20$ with $\epsilon = 8 * 10^{-7}$
  4. `sa_l_10_1swap`: Simulated Annealing with Logarithmic Schedule at $T_s = 10$ with $\epsilon = 1$

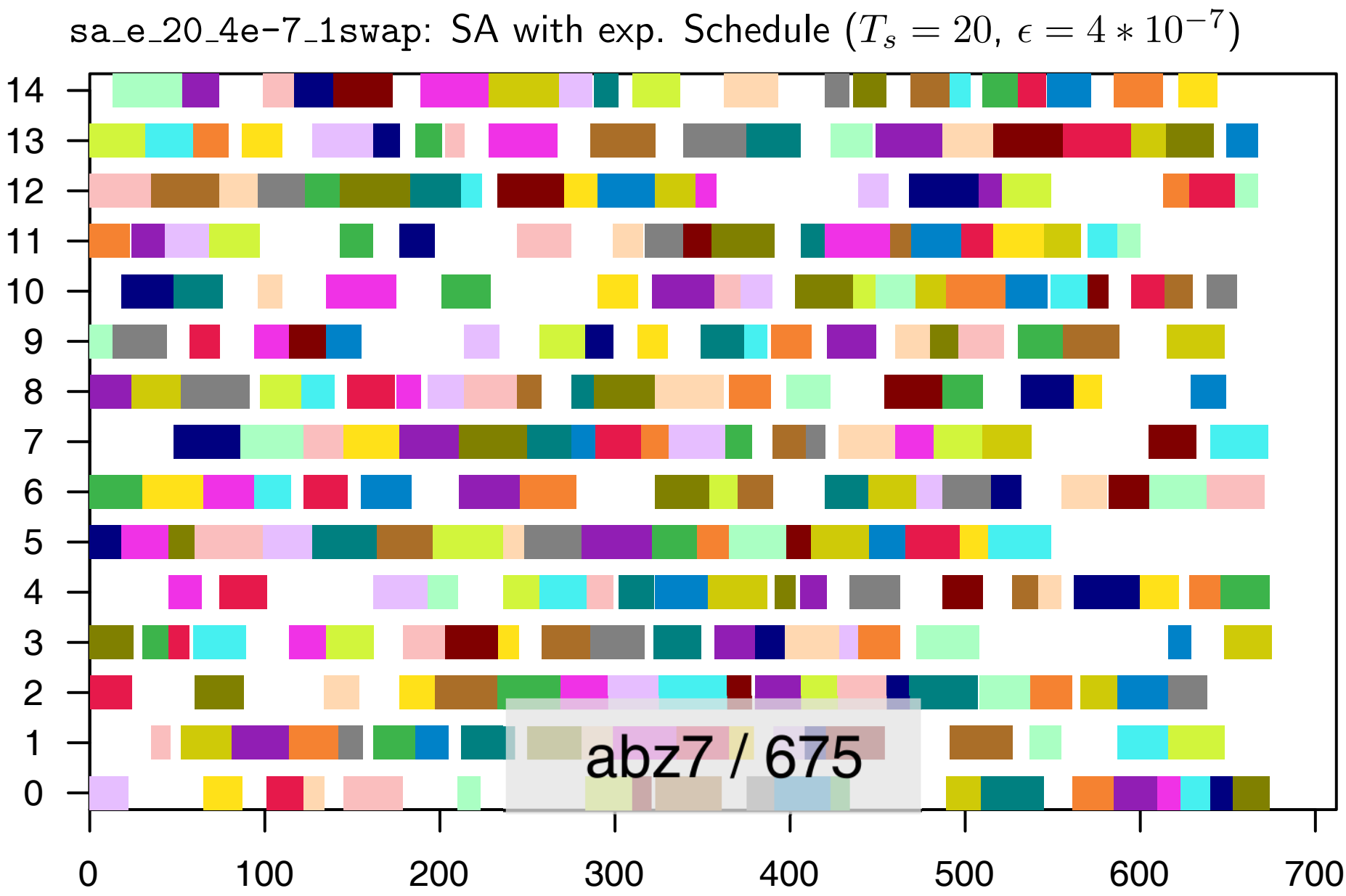
So what do we get?

		makespan				last improvement	
$\mathcal{I}$	algo	best	mean	med	sd	med(t)	med(FEs)
abz7	ea4096_1swap_5	680	712	712	10	20s	4'145'924
	sa_e_20_2e-7_1swap	663	673	672	5	92s	22'456'822
	sa_e_20_4e-7_1swap	**658**	674	675	5	55s	13'388'301
	sa_e_20_8e-7_1swap	663	675	675	6	36s	8'625'161
	sa_l_10_1swap	659	**672**	**671**	**4**	86s	21'271'077
la24	ea4096_1swap_5	945	976	975	15	4s	1'601'925
	sa_e_20_2e-7_1swap	938	**949**	**946**	**8**	27s	12'358'941
	sa_e_20_4e-7_1swap	**935**	949	**946**	9	16s	7'135'423
	sa_e_20_8e-7_1swap	**935**	951	950	8	9s	4'044'217
	sa_l_10_1swap	938	953	950	11	7s	3'210'824
swv15	ea4096_1swap_5	3413	3543	3539	66	169s	22'266'887
	sa_e_20_2e-7_1swap	2937	**2990**	**2988**	28	148s	21'949'073
	sa_e_20_4e-7_1swap	2941	2993	2993	**28**	128s	18'244'751
	sa_e_20_8e-7_1swap	**2936**	3000	3002	28	111s	16'029'528
	sa_l_10_1swap	2964	3021	3018	30	141s	21'252'052
yn4	ea4096_1swap_5	1034	1068	1068	18	37s	5'943'196
	sa_e_20_2e-7_1swap	973	**985**	**985**	**5**	113s	20'676'041
	sa_e_20_4e-7_1swap	**971**	987	986	7	68s	12'193'934
	sa_e_20_8e-7_1swap	972	988	988	7	58s	10'178'219
	sa_l_10_1swap	975	997	996	11	108s	19'850'143

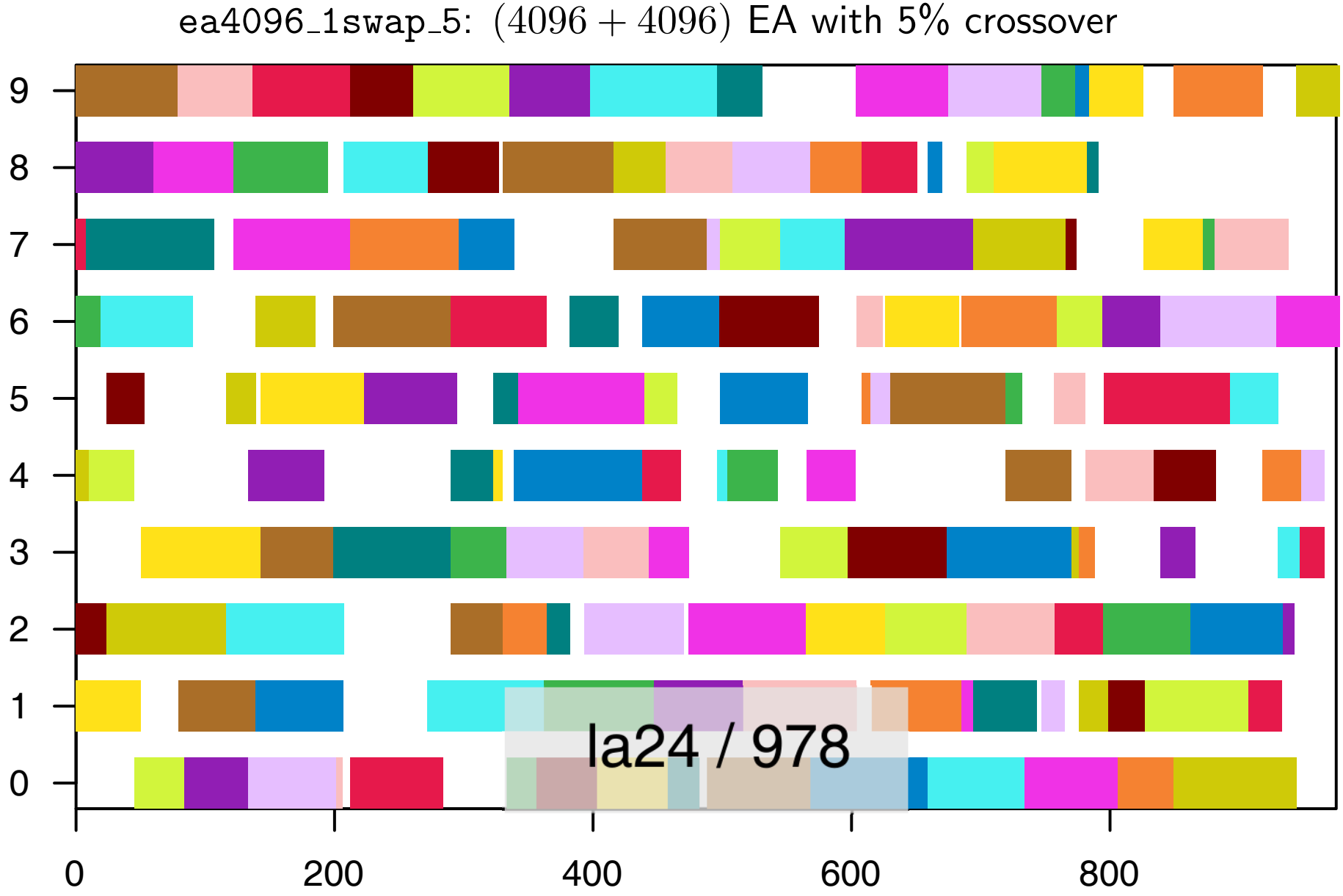
So what do we get?



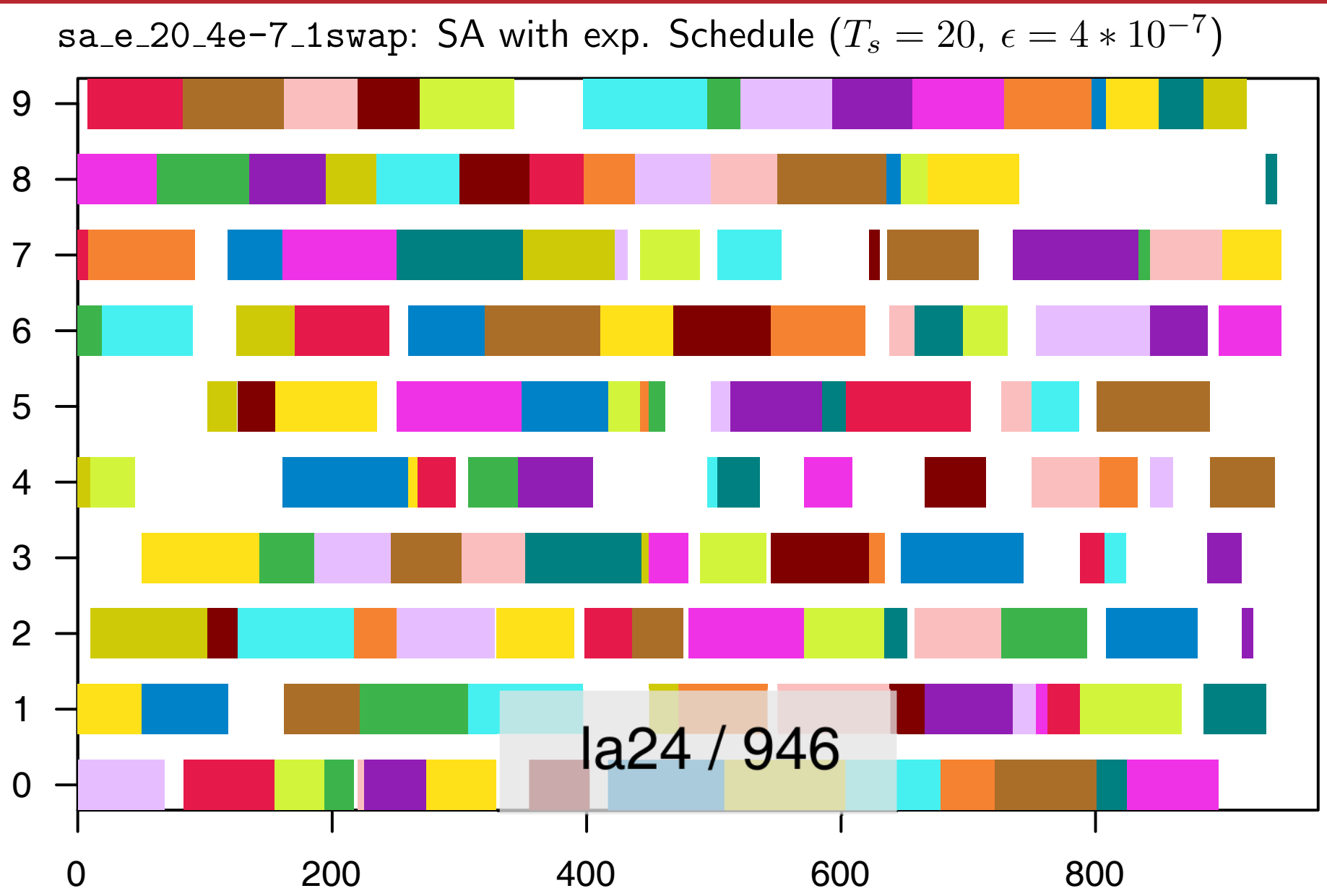
So what do we get?



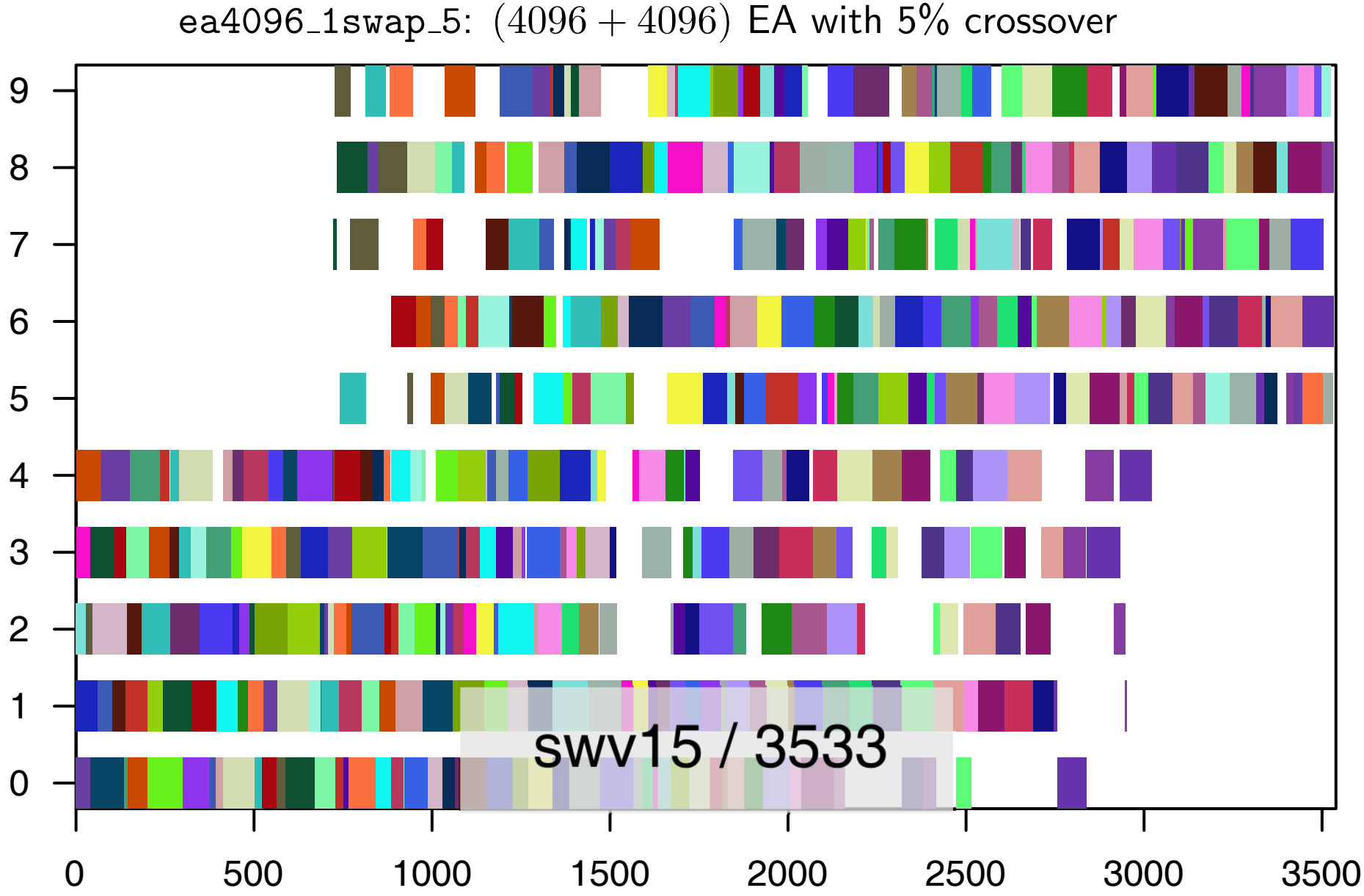
So what do we get?



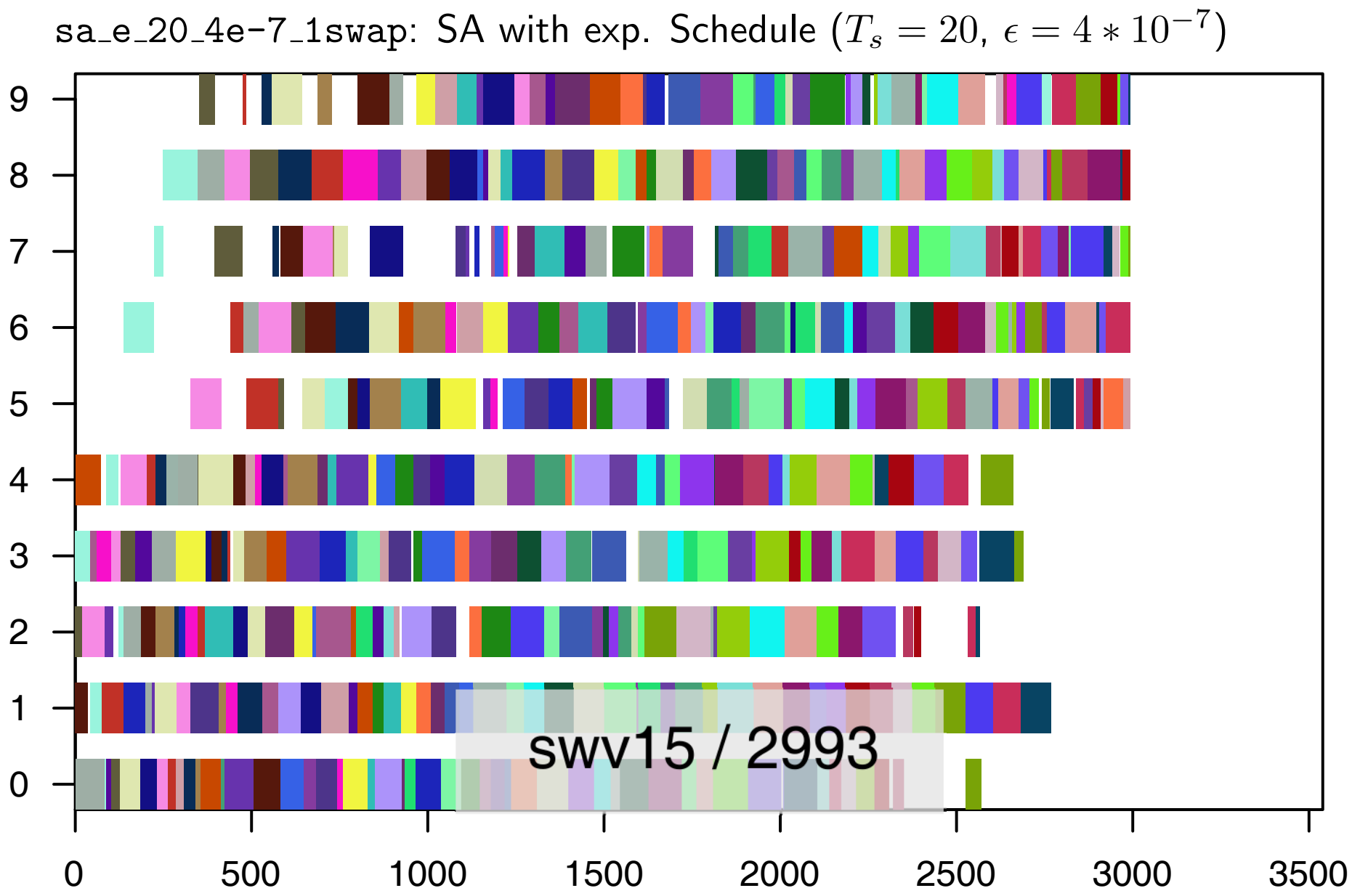
So what do we get?



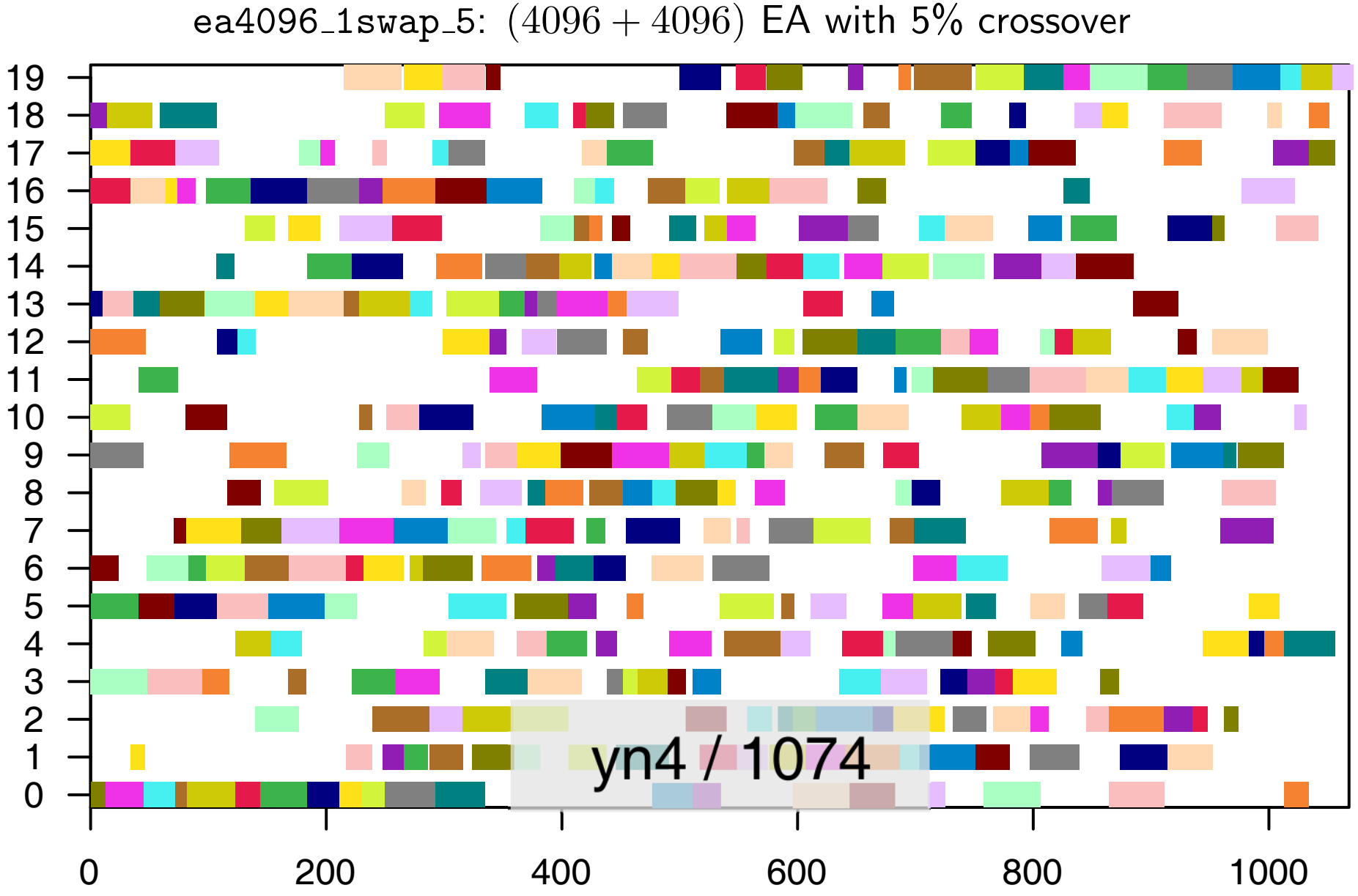
So what do we get?



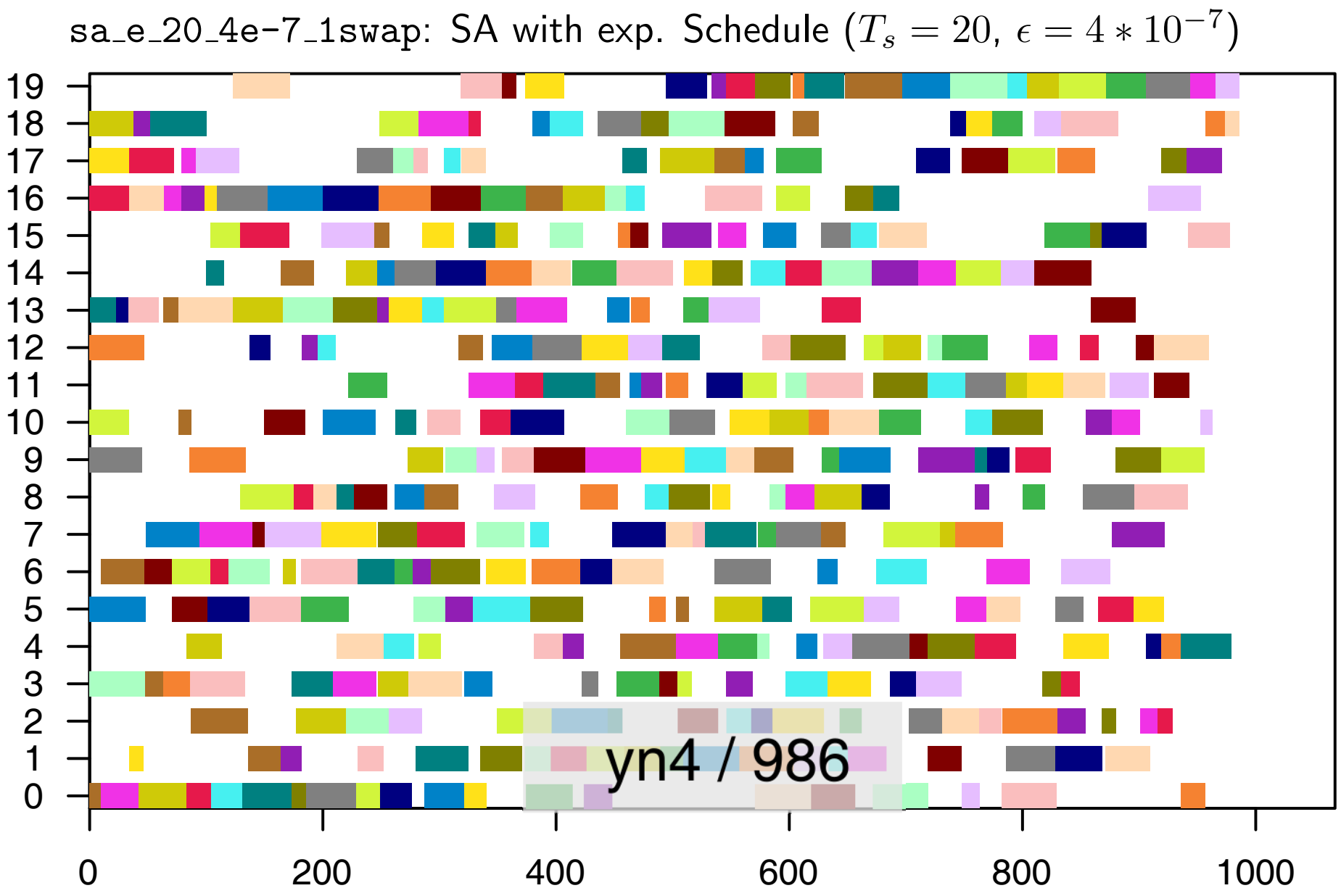
So what do we get?



So what do we get?



So what do we get?

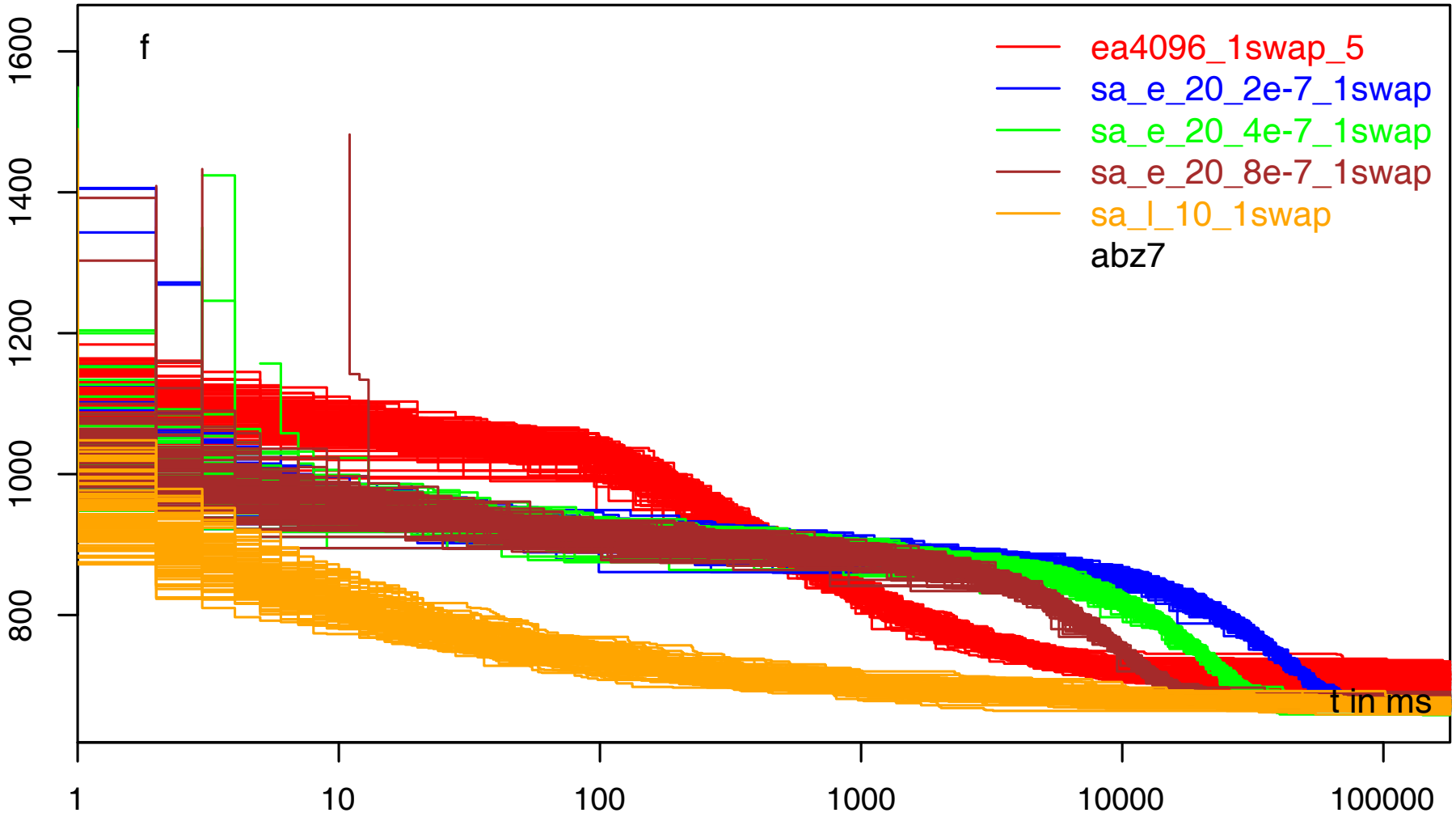


Progress over Time

What progress does the algorithm make over time?

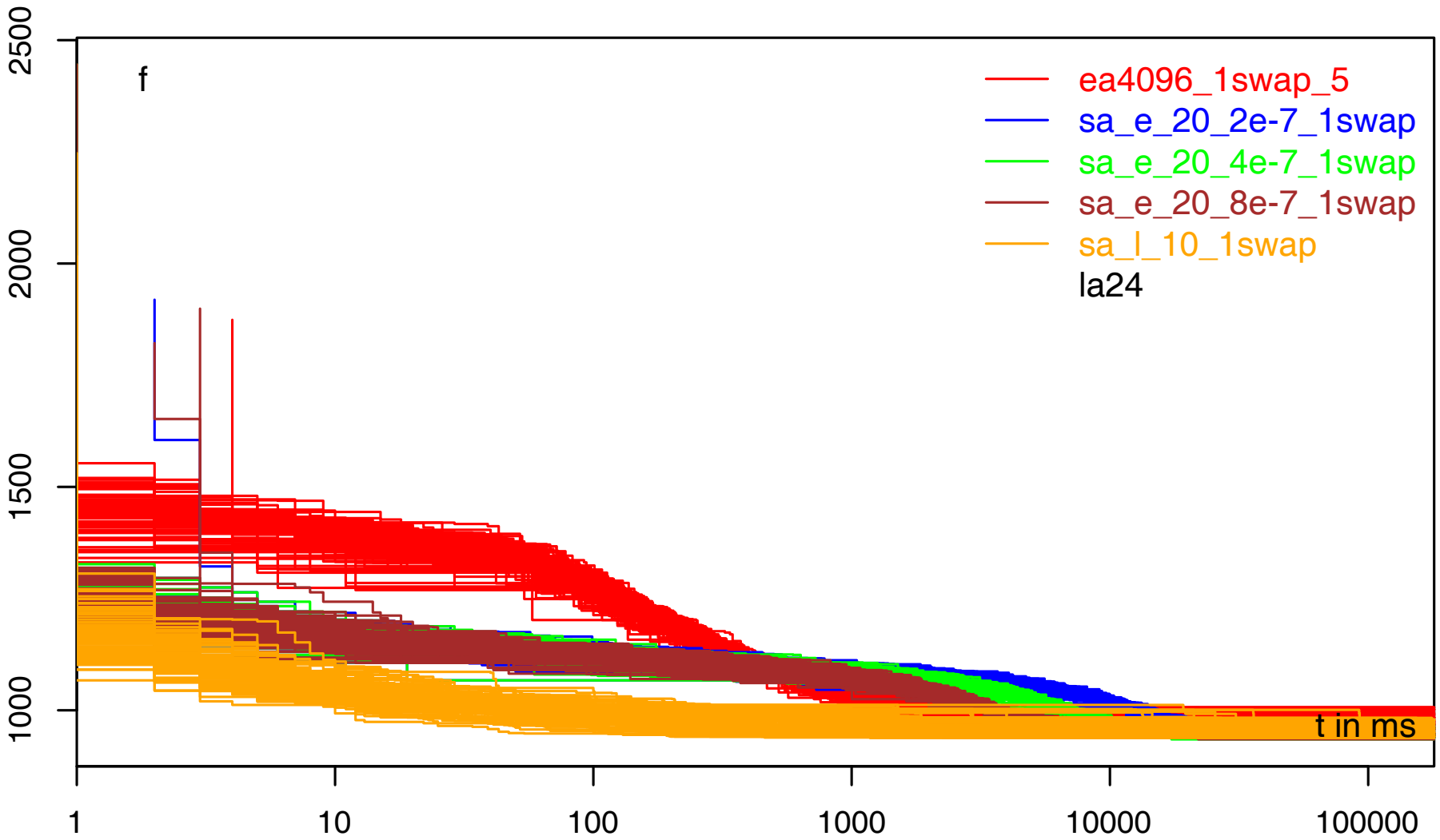
Progress over Time

What progress does the algorithm make over time?



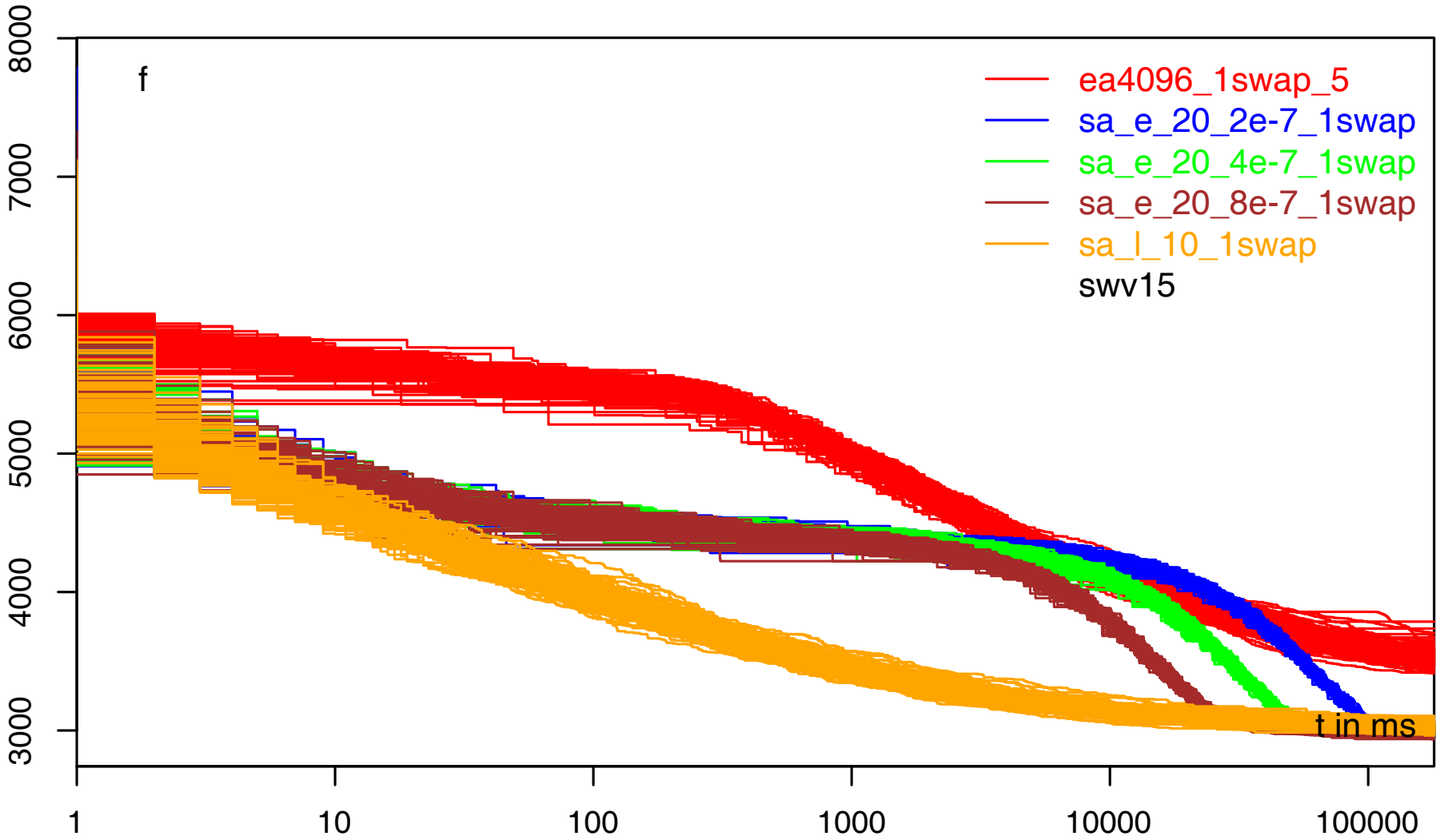
Progress over Time

What progress does the algorithm make over time?



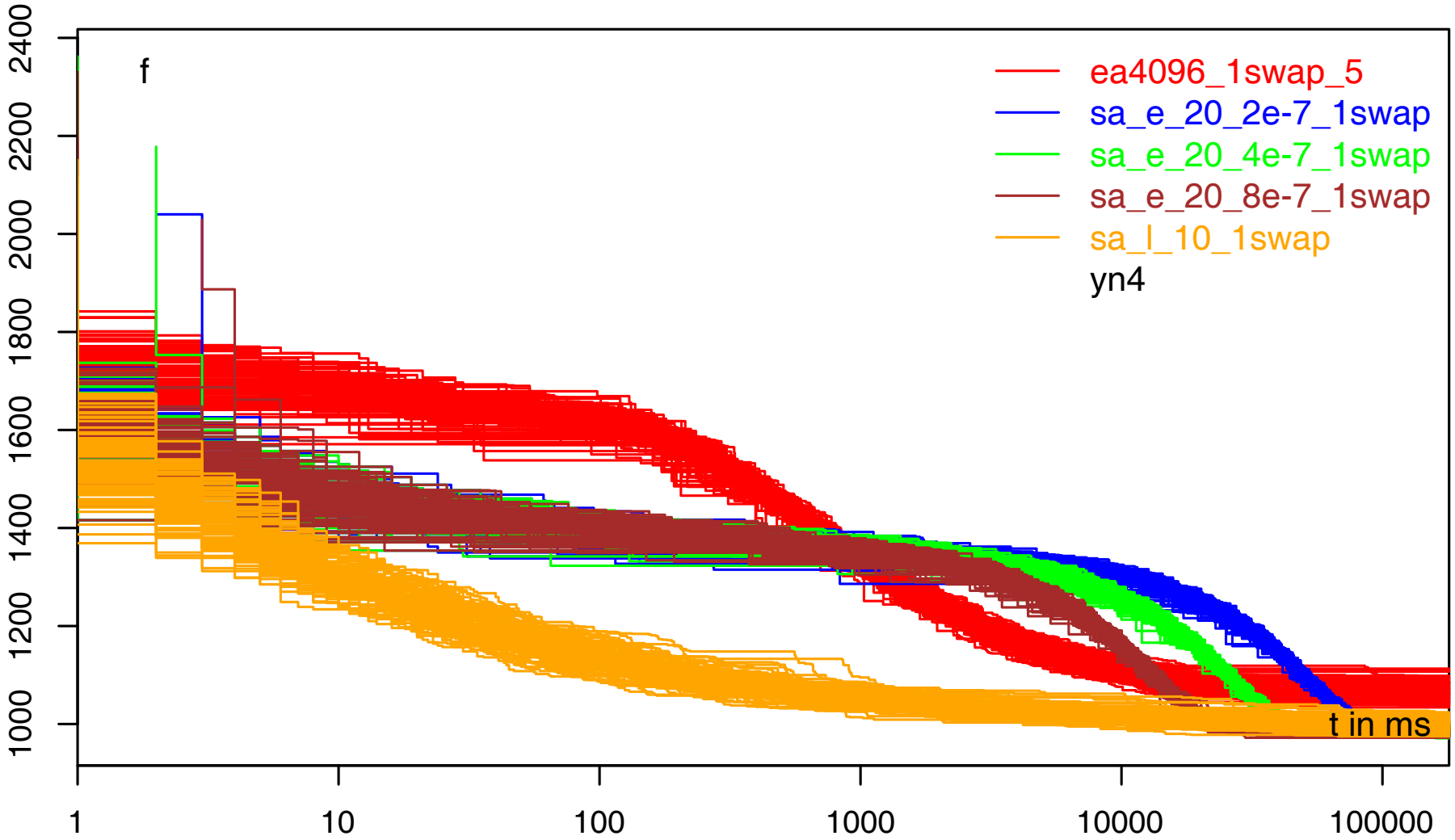
Progress over Time

What progress does the algorithm make over time?



Progress over Time

What progress does the algorithm make over time?



Summary

- Simulated Annealing outperformed all algorithms that we have tested before.

Summary

- Simulated Annealing outperformed all algorithms that we have tested before.
- But it also requires us to put more knowledge into the *configuration*

Summary

- Simulated Annealing outperformed all algorithms that we have tested before.
- But it also requires us to put more knowledge into the *configuration*:
  - We need to know approximately how many algorithm steps we can do within the computational budget.

Summary

- Simulated Annealing outperformed all algorithms that we have tested before.
- But it also requires us to put more knowledge into the *configuration*:
  - We need to know approximately how many algorithm steps we can do within the computational budget.
  - We need to know what "a bit worse" means in terms of the objective function.

Summary

- Simulated Annealing outperformed all algorithms that we have tested before.
- But it also requires us to put more knowledge into the *configuration*:
  - We need to know approximately how many algorithm steps we can do within the computational budget.
  - We need to know what "a bit worse" means in terms of the objective function.
  - We then need to determine a starting temperature T_s and a parameter ϵ to tune the temperature schedule accordingly.

Summary

- Simulated Annealing outperformed all algorithms that we have tested before.
- But it also requires us to put more knowledge into the *configuration*:
  - We need to know approximately how many algorithm steps we can do within the computational budget.
  - We need to know what "a bit worse" means in terms of the objective function.
  - We then need to determine a starting temperature T_s and a parameter ϵ to tune the temperature schedule accordingly.
- Perspective

Summary

- Simulated Annealing outperformed all algorithms that we have tested before.
- But it also requires us to put more knowledge into the *configuration*:
  - We need to know approximately how many algorithm steps we can do within the computational budget.
  - We need to know what "a bit worse" means in terms of the objective function.
  - We then need to determine a starting temperature T_s and a parameter ϵ to tune the temperature schedule accordingly.
- Perspective:
  - An Evolutionary Algorithm allows us to pick a behavior in between a hill climber and a random sampling algorithm by choosing a small or large population size.

Summary

- Simulated Annealing outperformed all algorithms that we have tested before.
- But it also requires us to put more knowledge into the *configuration*:
  - We need to know approximately how many algorithm steps we can do within the computational budget.
  - We need to know what "a bit worse" means in terms of the objective function.
  - We then need to determine a starting temperature T_s and a parameter ϵ to tune the temperature schedule accordingly.
- Perspective:
  - An Evolutionary Algorithm allows us to pick a behavior in between a hill climber and a random sampling algorithm by choosing a small or large population size.
  - The Simulated Annealing algorithm allows for a smooth transition of a random search behavior towards a hill climbing behavior over time.

Summary

- Simulated Annealing outperformed all algorithms that we have tested before.
- But it also requires us to put more knowledge into the *configuration*:
  - We need to know approximately how many algorithm steps we can do within the computational budget.
  - We need to know what "a bit worse" means in terms of the objective function.
  - We then need to determine a starting temperature T_s and a parameter ϵ to tune the temperature schedule accordingly.
- Perspective:
  - An Evolutionary Algorithm allows us to pick a behavior in between a hill climber and a random sampling algorithm by choosing a small or large population size.
  - The Simulated Annealing algorithm allows for a smooth transition of a random search behavior towards a hill climbing behavior over time.
  - Compared to the hill climber with restarts, it offers a "softer" way to escape from local optima which sacrifices less solution information.

谢谢

Thank you

Thomas Weise [汤卫思]
tweise@hfuu.edu.cn
http://iao.hfuu.edu.cn

Hefei University, South Campus 2
Institute of Applied Optimization
Shushan District, Hefei, Anhui,
China

Caspar David Friedrich, “Der Wanderer über dem Nebelmeer”, 1818
http://en.wikipedia.org/wiki/Wanderer_above_the_Sea_of_Fog

Bibliography

Bibliography I

1. Thomas Weise. *An Introduction to Optimization Algorithms*. Institute of Applied Optimization (IAO), Faculty of Computer Science and Technology, Hefei University, Hefei, Anhui, China, 2019-06-25 edition, 2018–2019. URL http://thomasweise.github.io/aitoa/. see also [6].
2. Scott Kirkpatrick, C. Daniel Gelatt, Jr., and Mario P. Vecchi. Optimization by simulated annealing. *Science Magazine*, 220 (4598):671–680, May 13, 1983. doi: 10.1126/science.220.4598.671. URL http://citeseer.ist.psu.edu/viewdoc/summary?doi=10.1.1.18.4175.
3. Vladimír Černý. Thermodynamical approach to the traveling salesman problem: An efficient simulation algorithm. *Journal of Optimization Theory and Applications*, 45(1):41–51, January 1985. doi: 10.1007/BF00940812. URL http://mkweb.bcgsc.ca/papers/cerny-travelingsalesman.pdf.
4. Dean Jacobs, Jan Prins, Peter Siegel, and Kenneth Wilson. Monte carlo techniques in code optimization. *ACM SIGMICRO Newsletter*, 13(4):143–148, December 1982. Proceedings of the 15th Annual Workshop on Microprogramming (MICRO 15), October 5–7, 1982, Palo Alto, CA, USA, New York, NY, USA: ACM.
5. Martin Pincus. Letter to the editor – a monte carlo method for the approximate solution of certain types of constrained optimization problems. *Operations Research*, 18(6):1225–1228, November–December 1970. doi: 10.1287/opre.18.6.1225.
6. Thomas Weise. *Global Optimization Algorithms – Theory and Application*. it-weise.de (self-published), Germany, 2009. URL http://www.it-weise.de/projects/book.pdf.
7. James C. Spall. *Introduction to Stochastic Search and Optimization*, volume 6 of *Estimation, Simulation, and Control – Wiley-Interscience Series in Discrete Mathematics and Optimizationn*. Wiley Interscience, Chichester, West Sussex, UK, April 2003. ISBN 0-471-33052-3. URL https://www.jhuapl.edu/ISSO/.
8. F. J. Humphreys and M. Hatherly. *Recrystallization and Related Annealing Phenomena*. Pergamon Materials Series. Amsterdam, The Netherlands: Elsevier Science Publishers B.V., 2004. ISBN 0080441645 and 9780080441641. URL http://books.google.de/books?id=52Gloa7HxGsC.
9. Nicholas Metropolis, Arianna W. Rosenbluth, Marshall Nicholas Rosenbluth, Augusta H. Teller, and Edward Teller. Equation of state calculations by fast computing machines. *The Journal of Chemical Physics*, 21(6):1087–1092, June 1953. doi: 10.1063/1.1699114. URL http://sc.fsu.edu/~beerli/mcmc/metropolis-et-al-1953.pdf.