The LION way: machine Learning for Intelligent OptimizatioN: a source of power for innovation in business

Roberto Battiti
University of Trento (Italy)

NUMTA2016
19 – 25 June 2016
Club Med Resort “Napitia”
Pizzo Calabro, Calabria, Italy

Driving forces in business creation and innovation

Automation

Google,facebook,twitter....

→ **Huge productivity increases**

1/10 people per revenue

→ **Fast adaptation/response, mass customization**

Data-driven models + Optimization

Automation needs meta-optimization and machine learning

Self-tuning: Method and parameters **selection, adaptation, configuration,** from building blocks

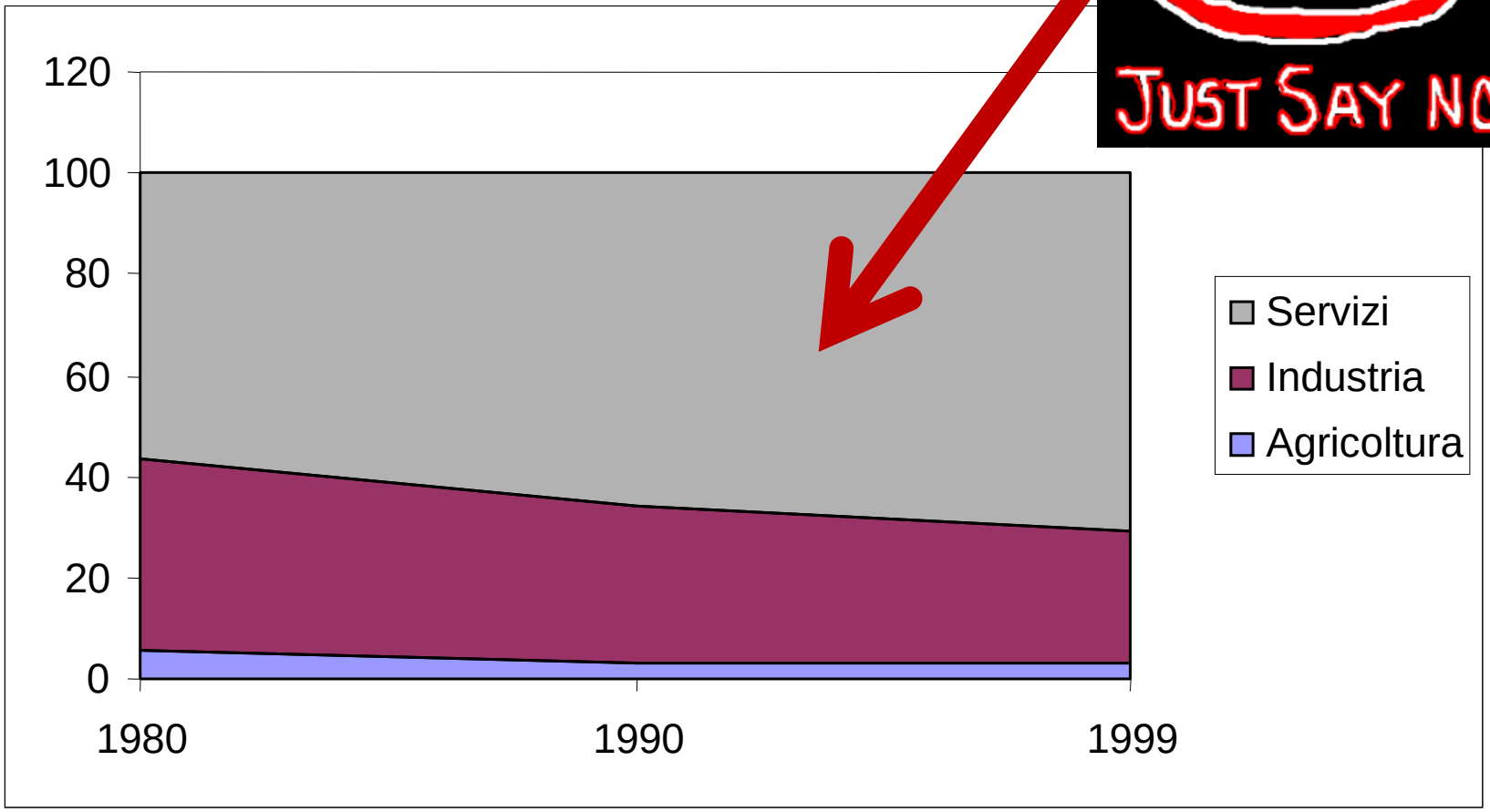
Learning the problem definition from data

Optimizing event locations...

Optimization is fine but ... What is the function?

Online learning!

A new revolution



(fonte http://www.worldbank.com/)

Optimization …in 2016

- **Basic optimization heuristics are becoming a commodity**

  More and more difficult to discover radically new techniques. Often **many** techniques. Subtle differences evaporate in real-world **noise**.

NOTE: **using new names for old building blocks** (local search, greedy constructions, diversification vs. intensification) **does not count as novelty!**

Metaheuristics the Metaphor Exposed

Kenneth Sorensen, Sep 2012, prev. Fred Glover

- jumps of frogs,
- refraction of light
- flowing of water to the sea,
- orchestra playing,
- sperm cells moving to fertilize an egg,
- spiraling movements of galaxies,
- colonizing behavior of empires,
- behavior of bats, birds, ants, bees, flies, and virtually every other species of insects

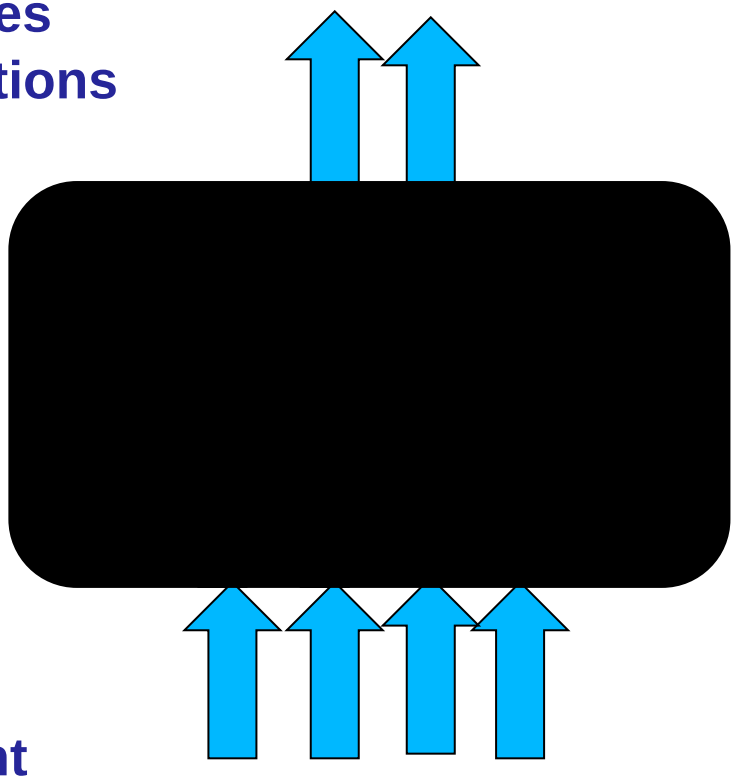
SA and GA started it all !

it seems that there is not a single natural or man-made process that cannot be used as a metaphor for yet another “novel“ optimization method.

Real word is dirty (black?)

Some posivite objectives (MOOP)
Combination not clear
Hidden objectives
Dynamic aspirations

No math formula
Maybe some high-level knowledge and intuition



Many inputs, noisy, some irrelevant

Learn !

Learn !

Learn !

Machine Learning !

Optimization: a tremendous power

Tapping and musik

- Still largely unexploited in most real-world contexts: standard optimization assumes a **function f(x)** to be minimized, …and **math** knowledge.
- function f(x) helps people to **concentrate on goals/objectives**, not on algorithms (on policies not on processes)

«Aortic radius minimizes dissipated power» -Pardalos

Try asking a manager

- BUT static f(x) does not exist in explicit form or is extremely difficult and costly to build by hand, and math knowledge is scarce.

Optimization: a tremendous power

- **Machine learning: learn f(x) from data (including from user feedback)**

- **Learning and Intelligent Optimization (LION): machine learning from data for optimization** which can be applied to complex, noisy, dynamic contexts.
- **ML to approximate f(x)** but also **to guide opt. process via *self-tuning*, both *offline* and *online***

- **Autonomy: more power directly in the hands of businesses**

Optimization → for Machine Learning

Source of power

Flexible model (with parameters **w**) How to pick **w?**

ErrorFunction E(w)

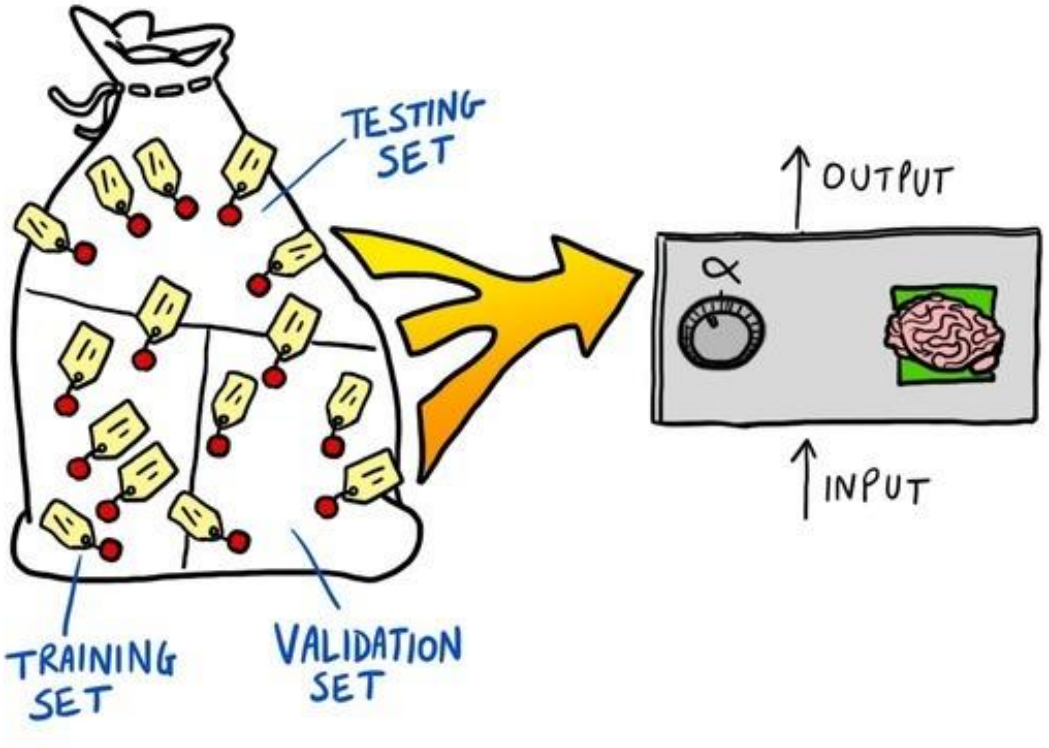
Learn by minimizing E(w) on training examples

...generalization

complicates a bit

MLP and Backpropagation

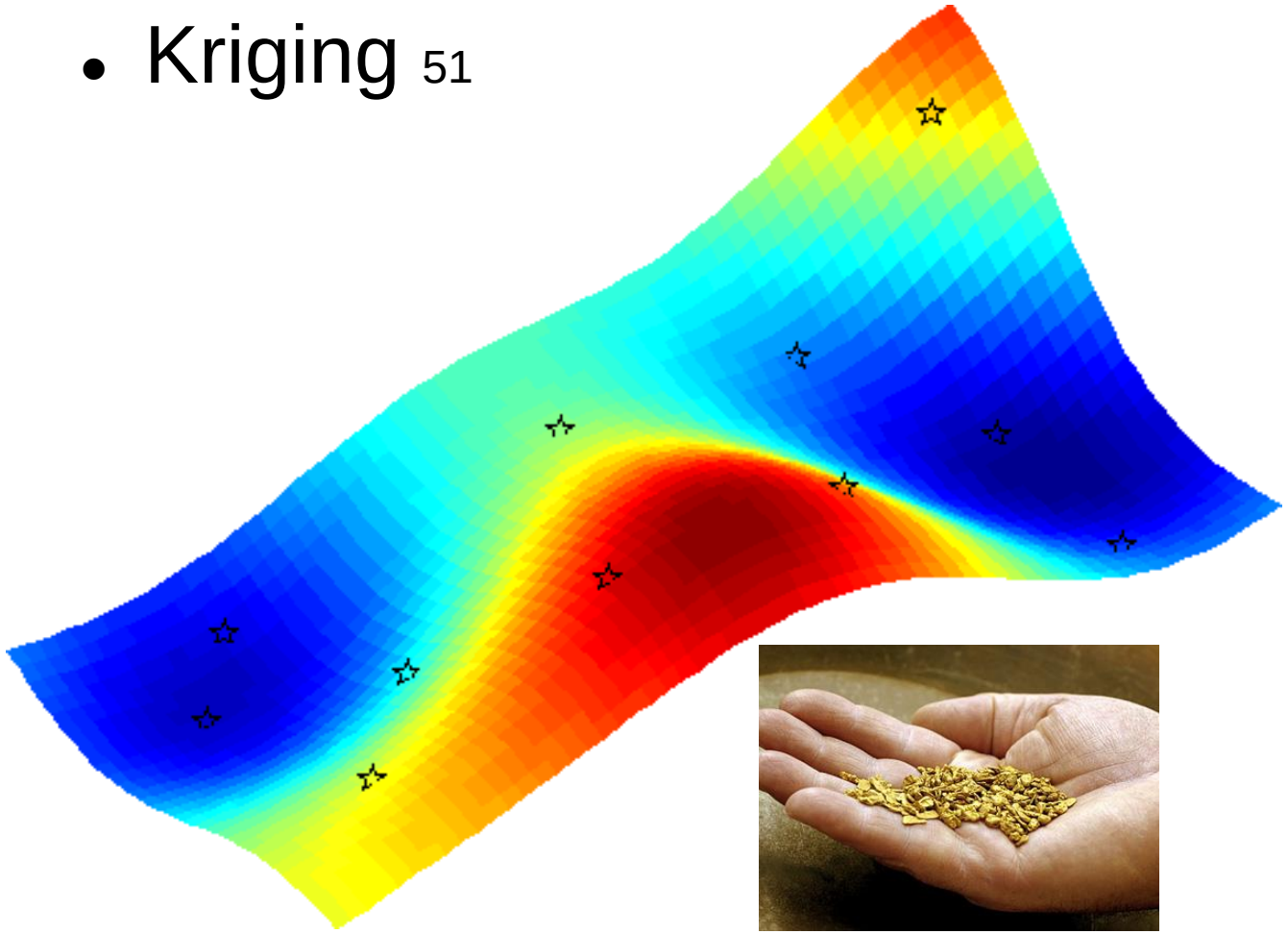
SVM ...



Machine Learning → for Optimization

Practical optimization is coslty ...f(x)

- Kriging 51



Danie Gerhardus Krige
(26 August 1919 – 3 March 2013)

Gaussian processes, Bayesian inference, splines, local models in continous optimization....

Angela Kunoth: «adaptive multi-scale»

Machine Learning → for Optimization

Guide optimization process →
by learning from previous search history

Reactive Search Optimization (RSO)

… suitable if single clear f(x)

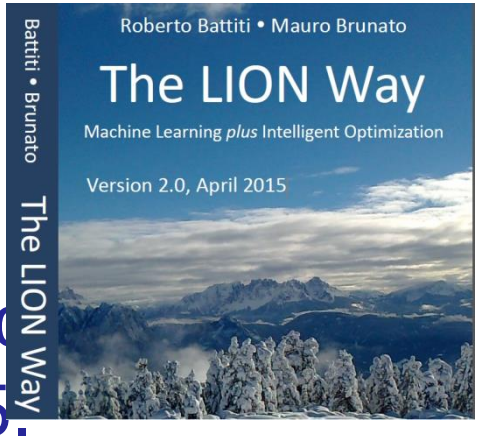
Build objectives →
by learning from decision maker

Brain-Computer Multi-Objective Optimization

… suitable if objectives are partially specified

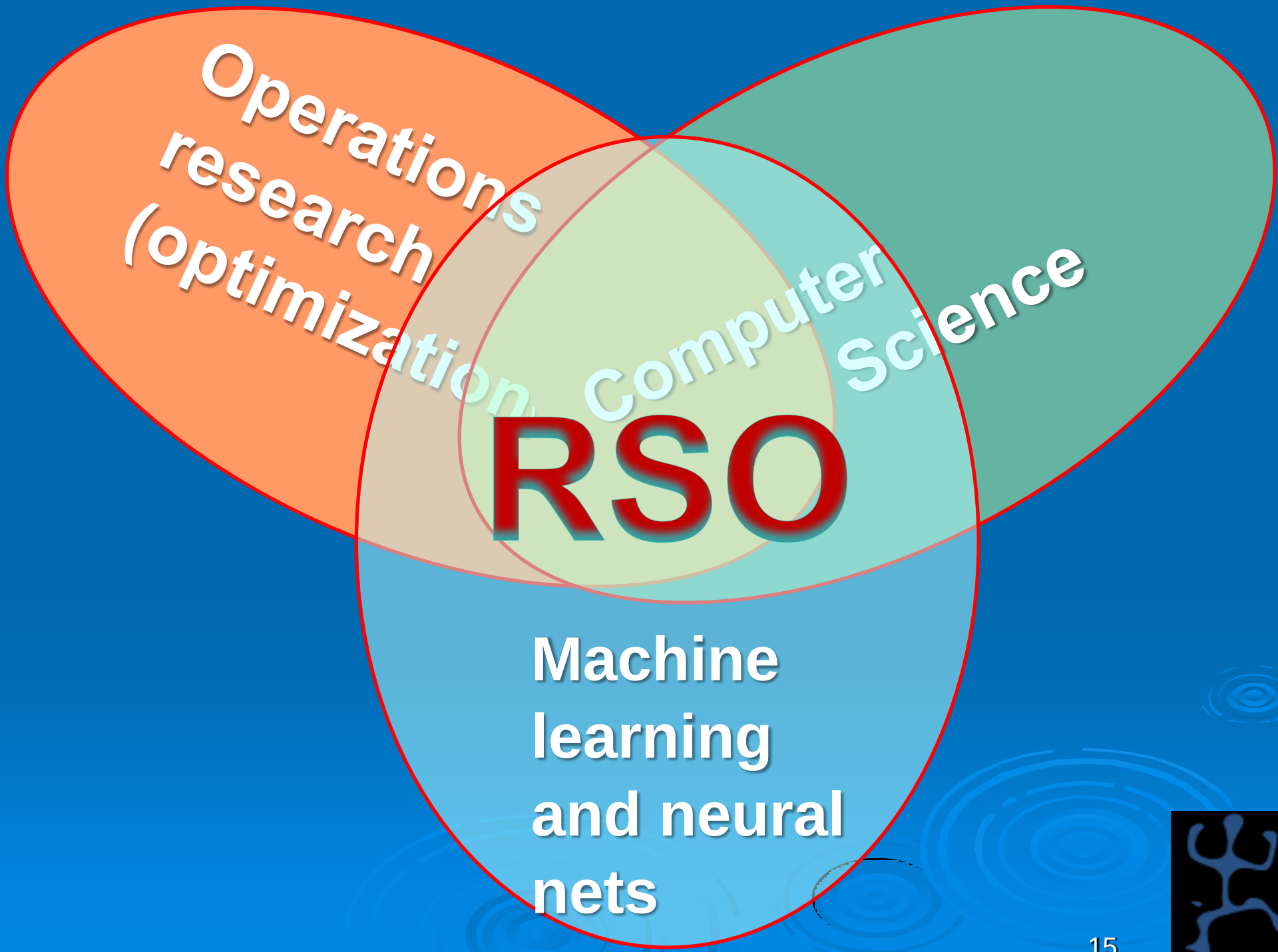
References

- http://rtm.science.unitn.it/~battiti/



- The LION way, Ver 2.0.
  Machine Learning plus Intelligent Optimizatio
  R. Battiti and M. Brunato, LIONlab, Apr 2015.

- Reactive Search and Intelligent Optimization,
  R. Battiti, M. Brunato and F. Mascia
  Springer Verlag, 2008


Operations research (optimization)
Computer Science
RSO
Machine learning and neural nets

The role of the user

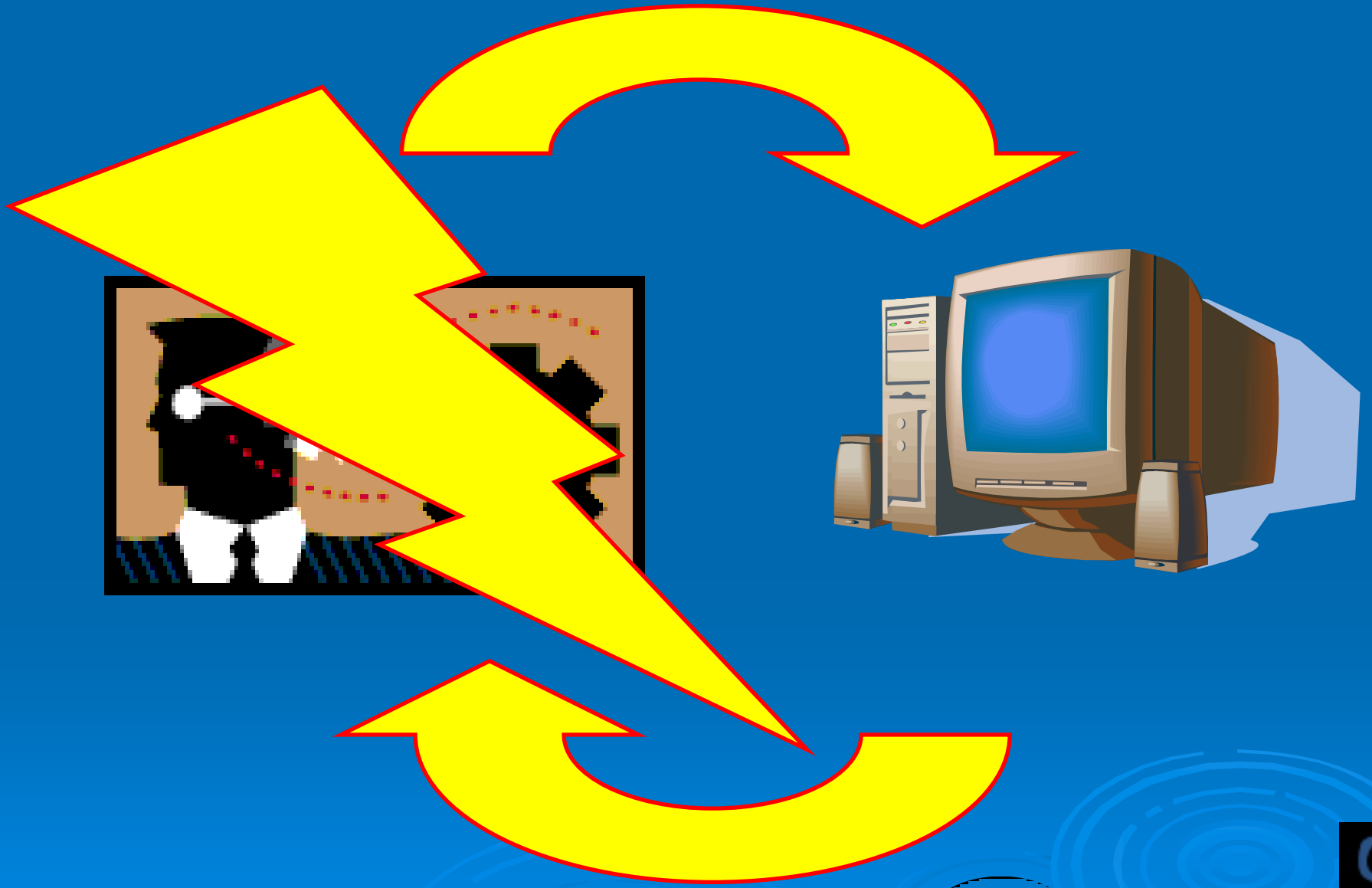
- **choices and free parameters**

  Algorithm(T)

- **the user as a crucial learning component** ("trial and error")

- **Parameter tuning is a typical "learning" process** where experiments are designed, with the support of statistical estimation (parameter identification) tools.

The role of the user



Automated tuning through machine learning

- **Automation**. The time-consuming tuning phase is now substituted by an automated process.
- **Complete and unambiguous documentation**. The algorithm becomes self-contained: its quality can be judged independently from the designer.

Complexity is shifted
Final user → algorithm developer
Reactive search optimization needs proactive researcers

Different from Markov process

$$Y \leftarrow \text{NEIGHBOR}(\, N(X^{(t)}) \,)$$

$$X^{(t+1)} = \begin{cases} Y & \text{if } f(Y) < f(X^{(t)}) \\ Y & \text{with probability } e^{-\Delta f/T} \text{, } X^{(t)} \text{ otherwise} \end{cases}$$

Simulated Annealing

$$T_k \geq \frac{\Gamma}{\log(k + k_0)}$$

$$\lim_{k \to \infty} \mathbb{P}\{X^{(k)} \in \mathcal{S}^*\} = 1$$

- Asymptotic convergence is irrelevant
- Slow “speed of convergence” to the stationary distribution... Complete enumeration can be faster!

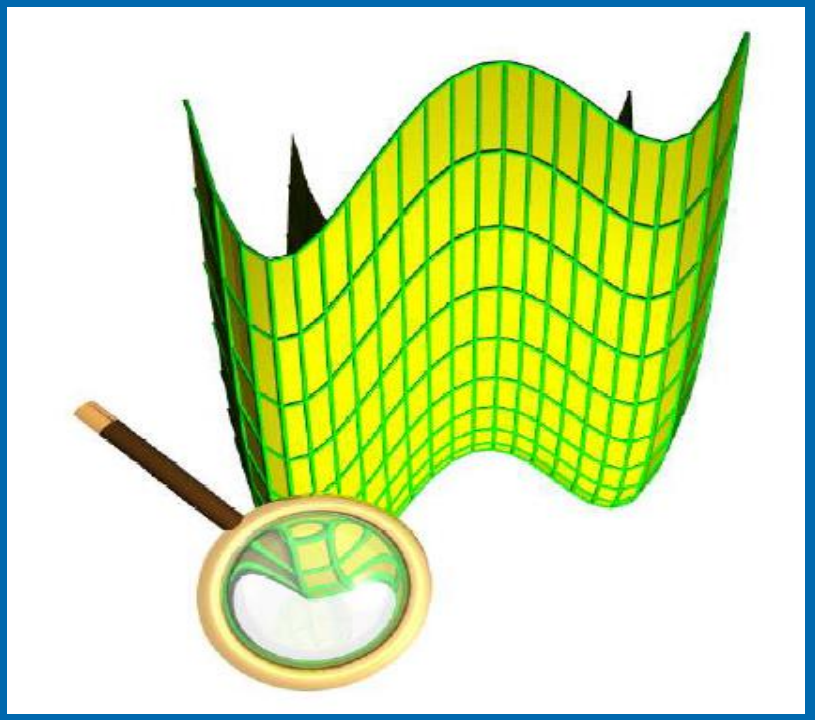
Reactive Search Optimization

integration of online machine learning techniques for local search heuristics.

The word ***reactive*** hints at a ready response to events *during* the search through an internal online feedback loop for the *self-tuning* of critical parameters.

Biological systems are highly adaptive; they use signals coming in from receptors and sensors to fine-tune their functioning. Adaptivity is a facet of the **reactivity** of such systems.

On-line tuning



- Take into account:

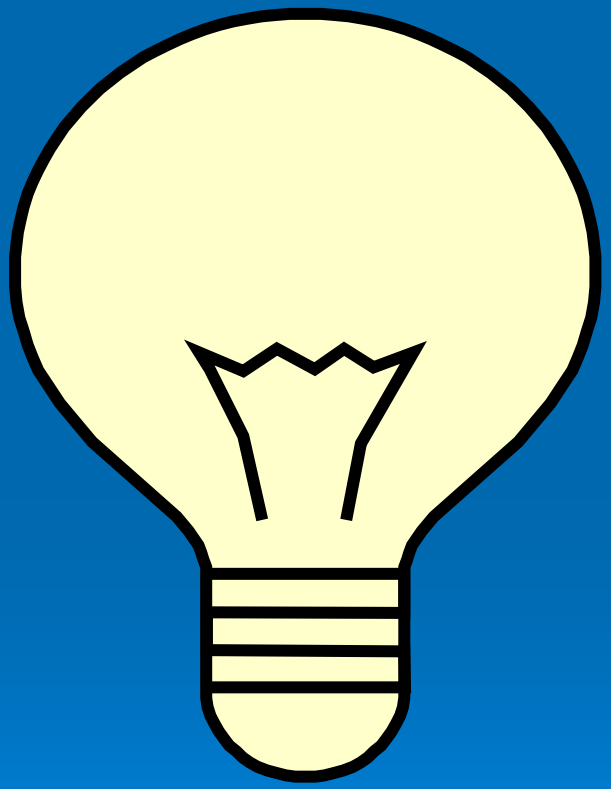
1. Problem-dependent
2. Task-dependent
3. Local properties in configuration space (see local search), parameters are dynamically tuned based on optimization state and previous history

RSO applied: intensification or diversification?

Diversification requires effort

It is a good morning exercise for a research scientist to discard a pet hypothesis every day before breakfast. It keeps him young.
Konrad Lorenz

A counter-example



An example: reactive prohibition-based local search

- X is the search space

0010110001000

- Neighborhood

$$N(X^{(t)}) = \{X \in \mathcal{X} \text{ such that } X = \mu_i \circ X^{(t)}, i = 0, \ldots M\}$$

- Search trajectory $X^{(0)}, \ldots, X^{(t+1)}$

$$Y \leftarrow \text{BEST-NEIGHBOR}(\, N(X^{(t)}) \,)$$

$$X^{(t+1)} = \begin{cases} Y & \text{if } f(Y) < f(X^{(t)}) \\ X^{(t)} & \text{otherwise (search stops)} \end{cases}$$

Prohibition-based local search

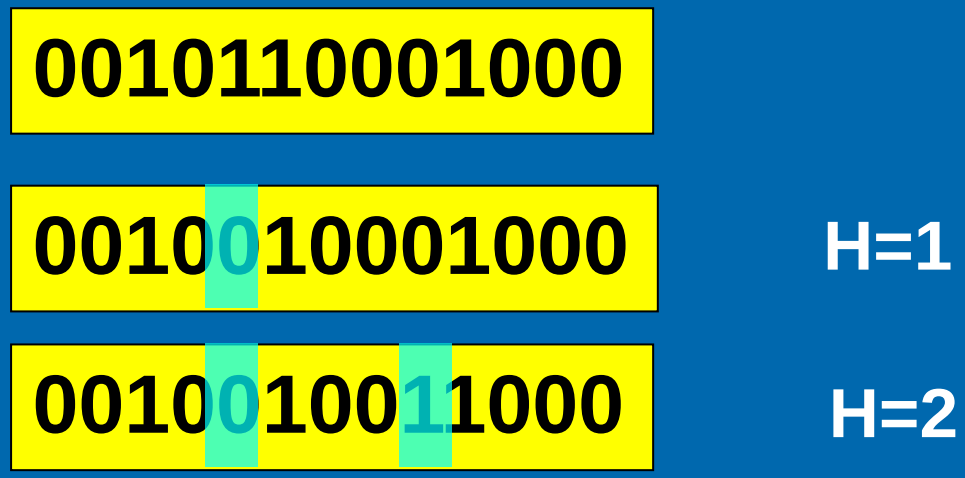
- Local search leads to local minima
- What next?
  - (random) restart
  - try to use knowledge accumulated during the previous searches *(learn !)*
    - … escape from previously visited basins of attraction around a local minimizer (diversification)
    - simple diversification through prohibitions

Prohibition-based local search (2)

- Prohibition-based: history
  - Steiglitz Weiner- *denial* strategy for TSP (once common features are detected in many suboptimal solutions, they are *forbidden*) (opposite to *reduction* strategy: all edges that are common to a set of local optima are fixed)
  - Lin-Kernighan for graph partitioning
  - Tabu Search (Fred Glover)
  - Steepest Ascent Mildest Descent (Hansen – Jaumard)

Prohibition-based local search (3)

- diversification through prohibitions



Fundamental relationship between **prohibition and diversification** (Battiti, 1999)

- The Hamming distance H between a starting point and successive point along the trajectory is strictly increasing for $T+1$ steps.

$$H(X^{(t+\Delta t)}, X^{(t)}) = \Delta t \quad \text{for} \quad \Delta t \leq T+1$$

- The minimum repetition interval R along the trajectory is $2(T+1)$.

$$X^{(t+R)} = X^{(t)} \;\Rightarrow\; R \geq 2(T+1)$$

Some forms of Tabu Search

- Allowed neighbors

$$N_A(X^{(t)}) \subseteq N(X^{(t)})$$

- Discrete dynamical system

$$\begin{aligned} X^{(t+1)} &= \textsc{Best-Neighbor}\left(N_A(X^{(t)})\right) \\ N_A(X^{(t+1)}) &= \textsc{Allow}\left(N(X^{(t+1)}), X^{(0)}, \ldots, X^{(t+1)}\right) \end{aligned}$$

Tabu Search: Prohibition Mechanisms

- Strict-TS

$$N_A(X^{(t+1)}) = \{X \in N(X^{(t+1)}) \text{ s. t. } X \notin \{X^{(0)}, ..., X^{(t+1)}\}\}$$

- Fixed-TS

$$N_A(X^{(t)}) = \{X = \mu \circ X^{(t)} \text{ s. t. } \textsc{LastUsed}(\mu^{-1}) < (t - T)\}$$

- Reactive-TS

? Are the dynamical systems comparable ?

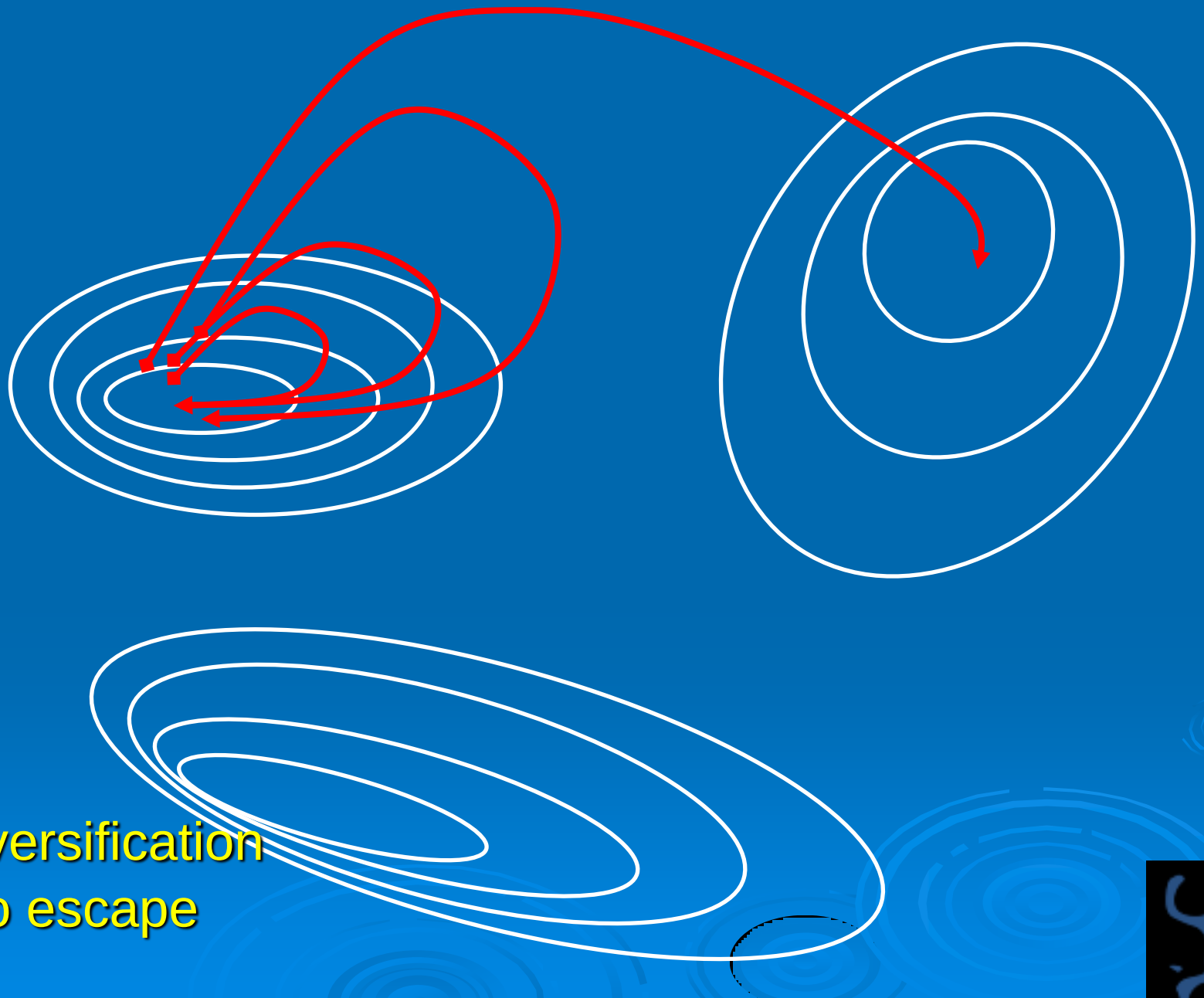
? Or qualitative differences ?

Distinguish policies from mechanisms

Issues in prohibition-based search

- Tuning of T (offline vs. reactive/online)
- Appropriate data structures for storing and accessing search history
- Robustness for a variety of applications

Reactive Prohibition-based search



Minimal diversification
sufficient to escape

Reactive prohibitions

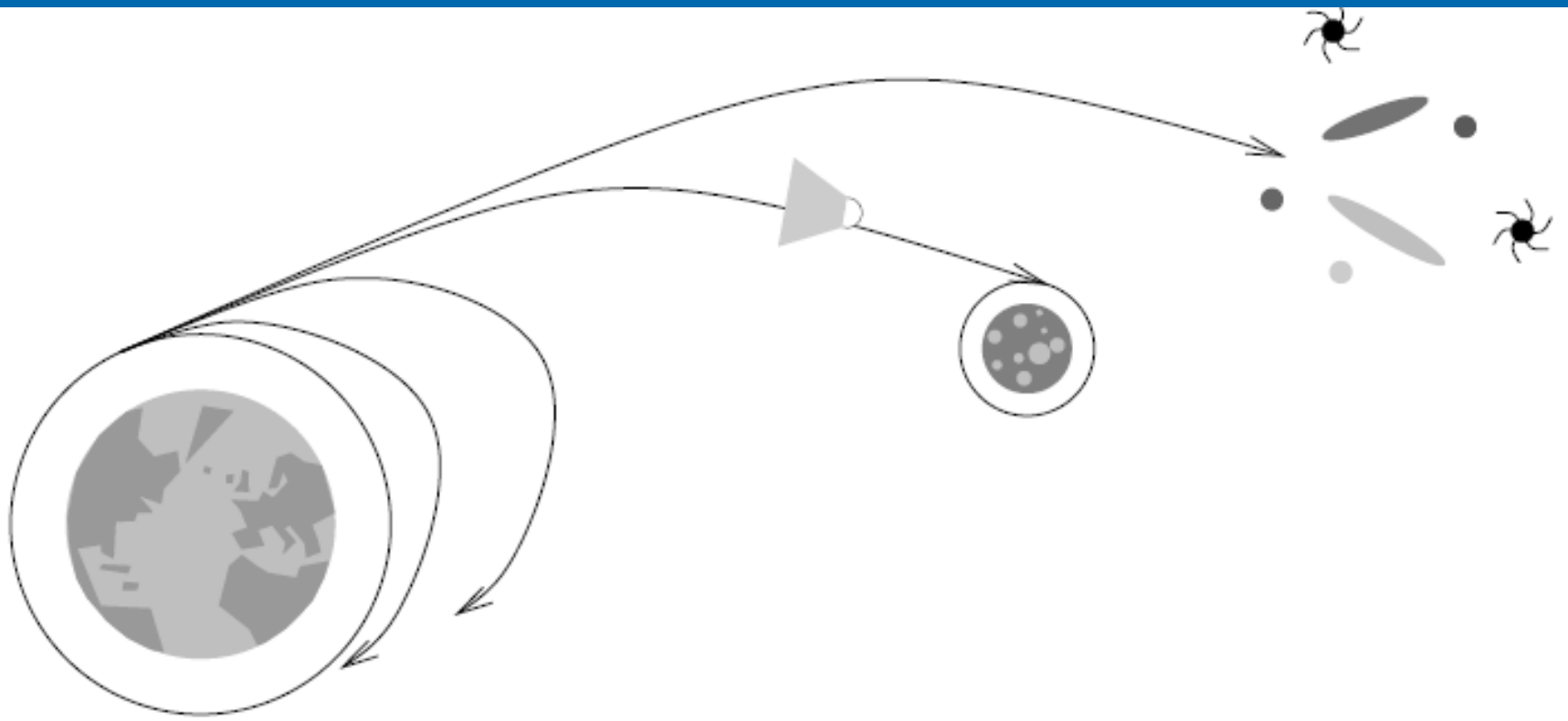
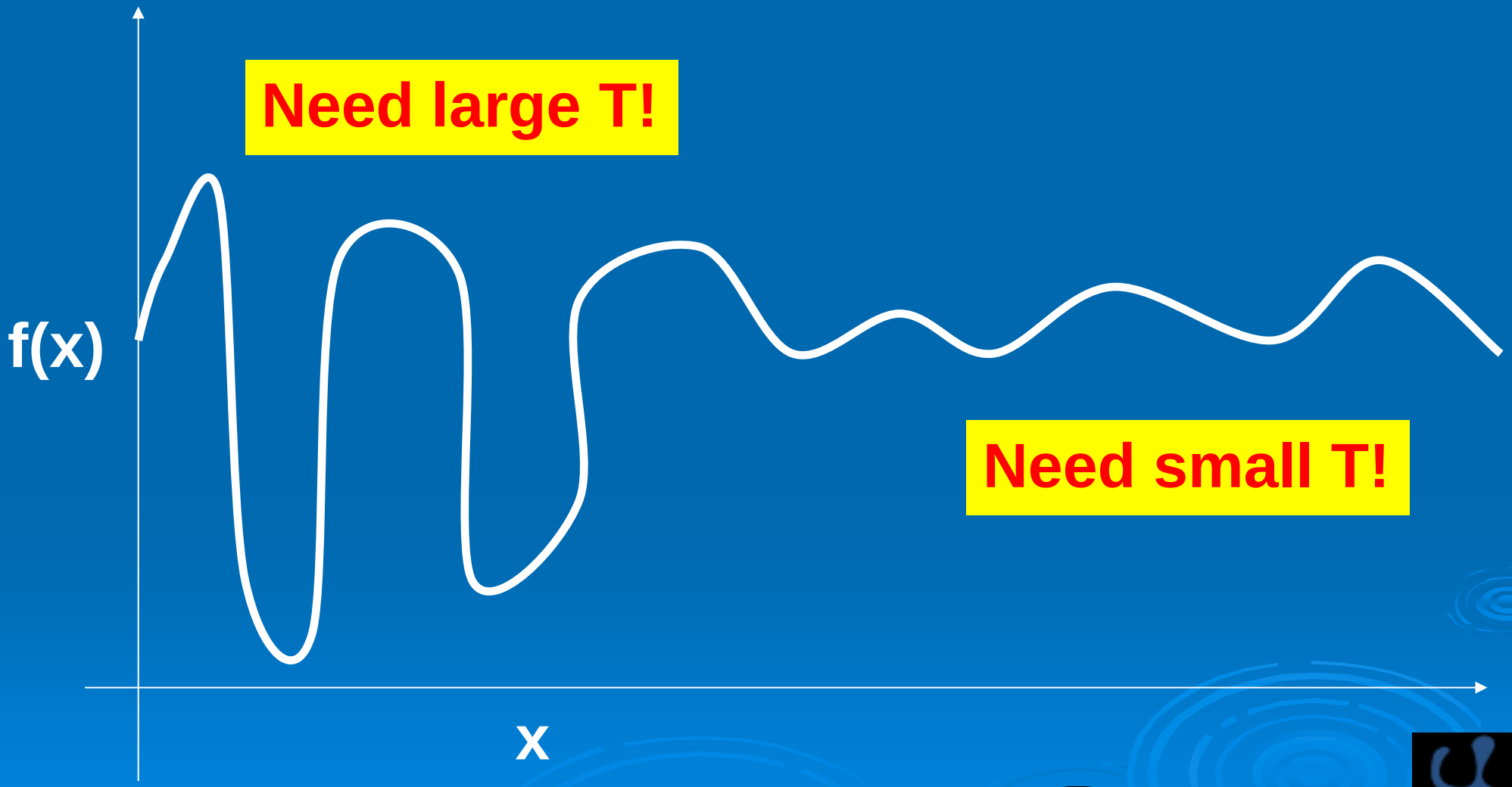


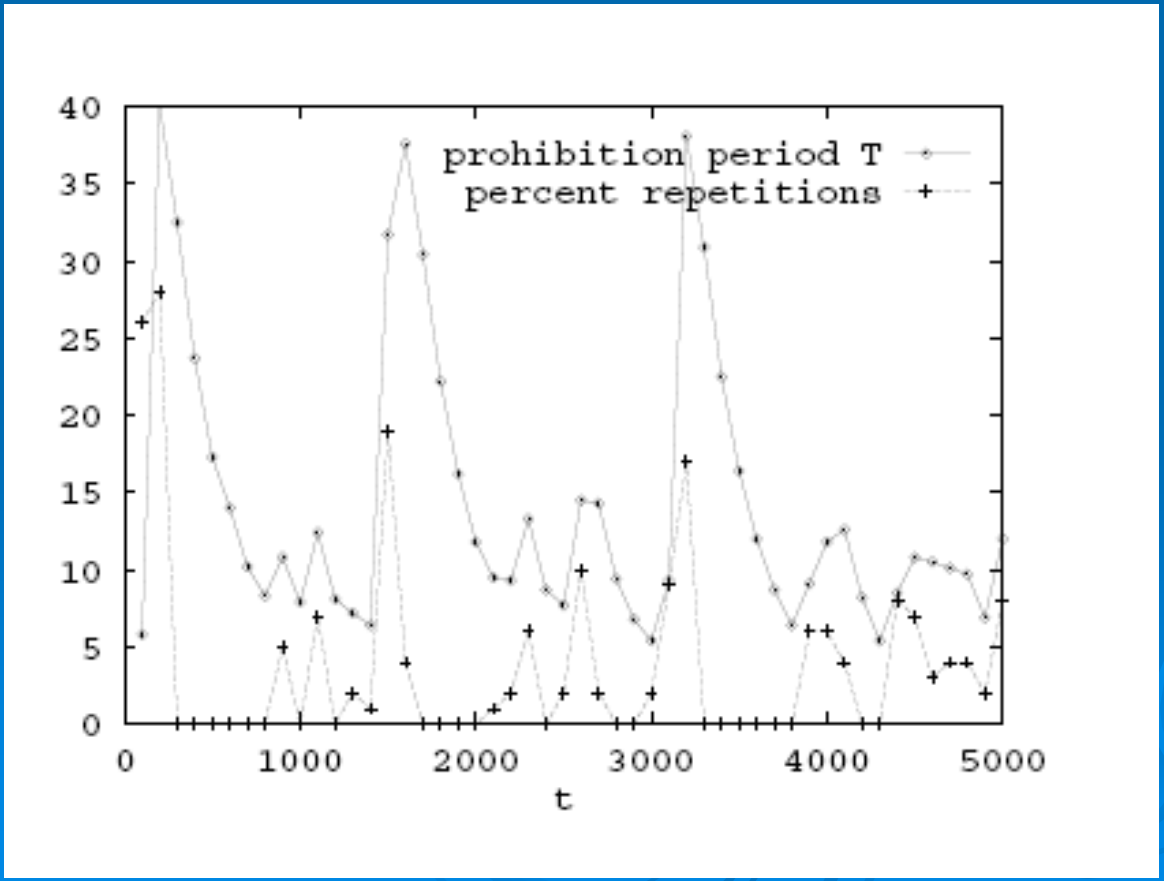
Figure 4.3: Applying the minimum diversification needed to escape the local attractor. Top: if too little diversification is applied, the trajectory falls back to the same minimum; if diversification is excessive, nearby minima may not be visited. Bottom: the same concept applied to space travel: the right amount of diversification (plus ballistic calculations) allows the capsule to reach a suitable moon orbit; too little and the capsule falls back, too much and the target is missed.

Motivations for a dynamic T



Self-adjusted T

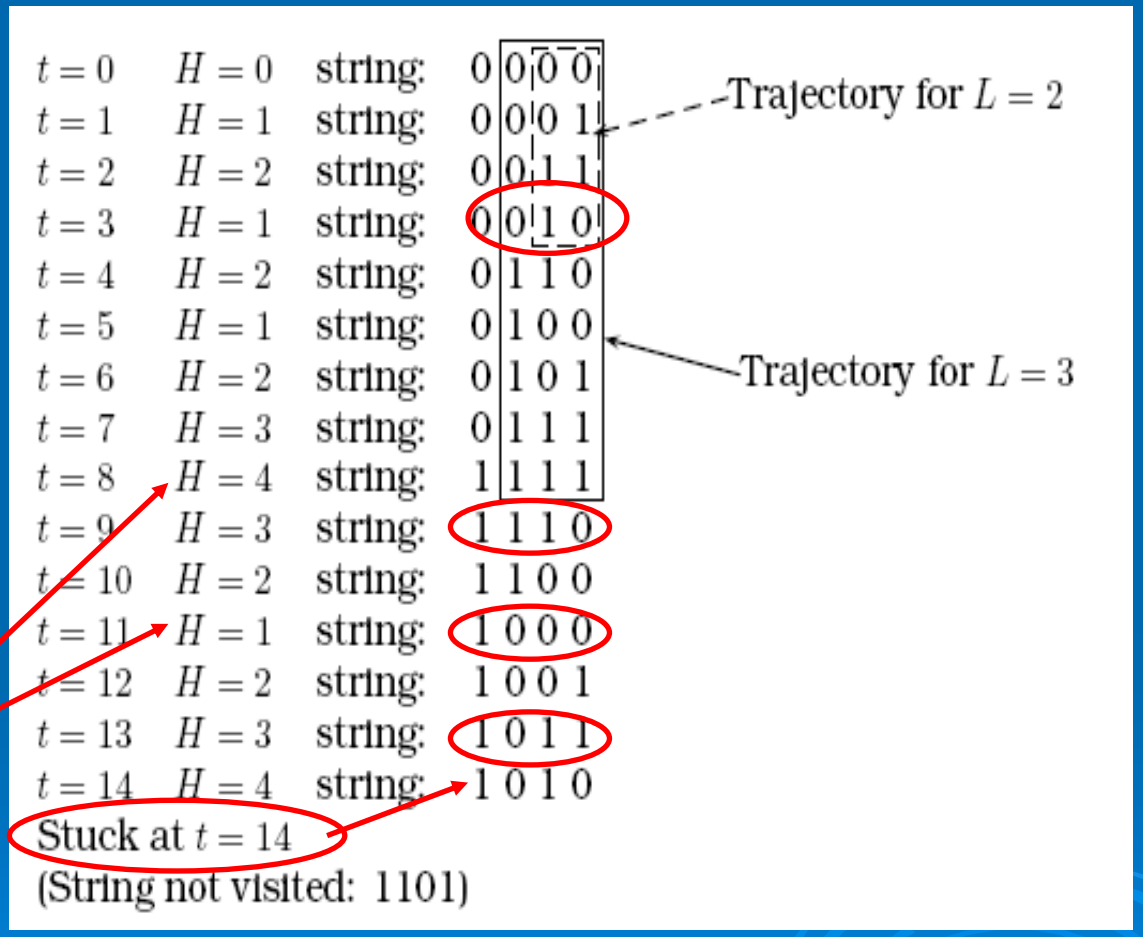
- T=1 at the beginning
- Increase T if evidence of *entrapment*
- Decrease T if evidence disappears



How to escape from an attractor

- Cost= Hamming distance from 00000
- Strict-TS

$$H(t) \leq \lfloor \log_2(t) \rfloor + 1$$



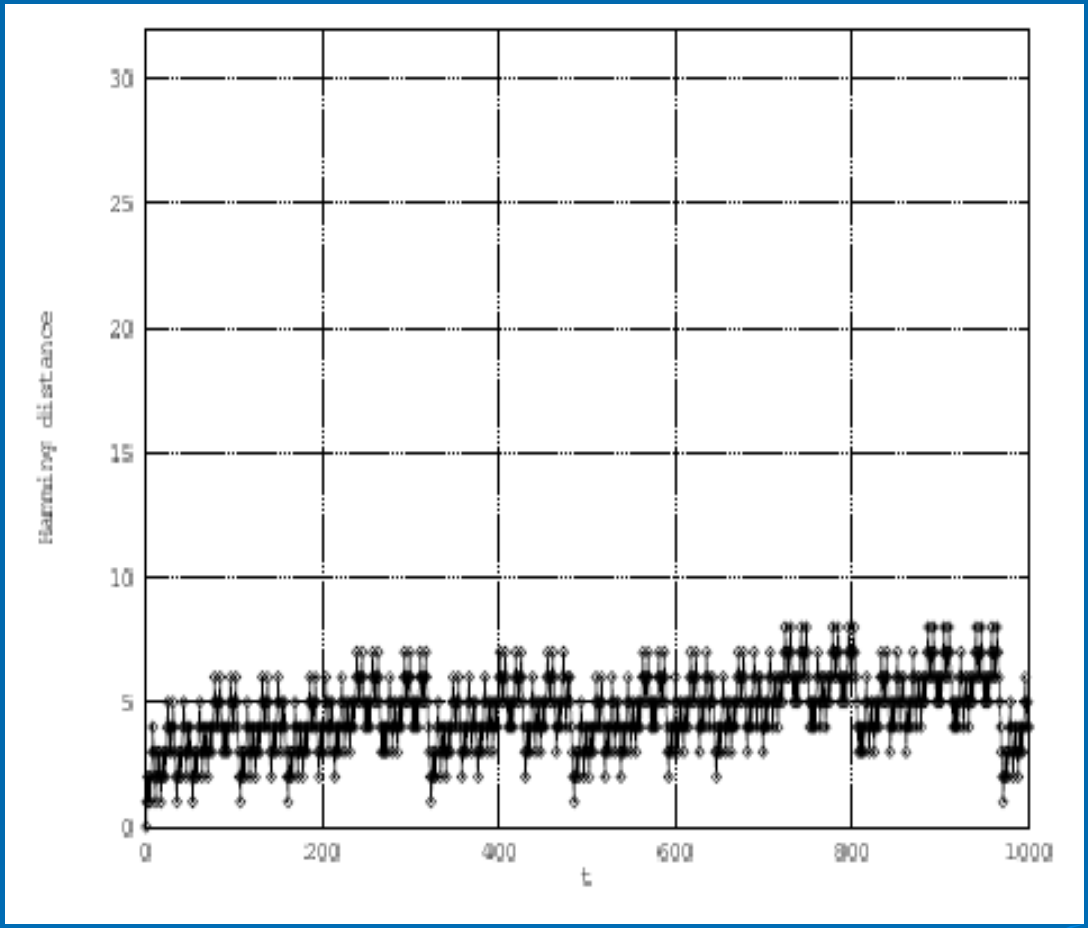
Non so intensifying…

How to escape from an attractor

- Strict-TS

$$C_H = \sum_{i=0}^{H} \binom{L}{i}$$

$$C_H >> 2^H, \text{ if } H << L$$

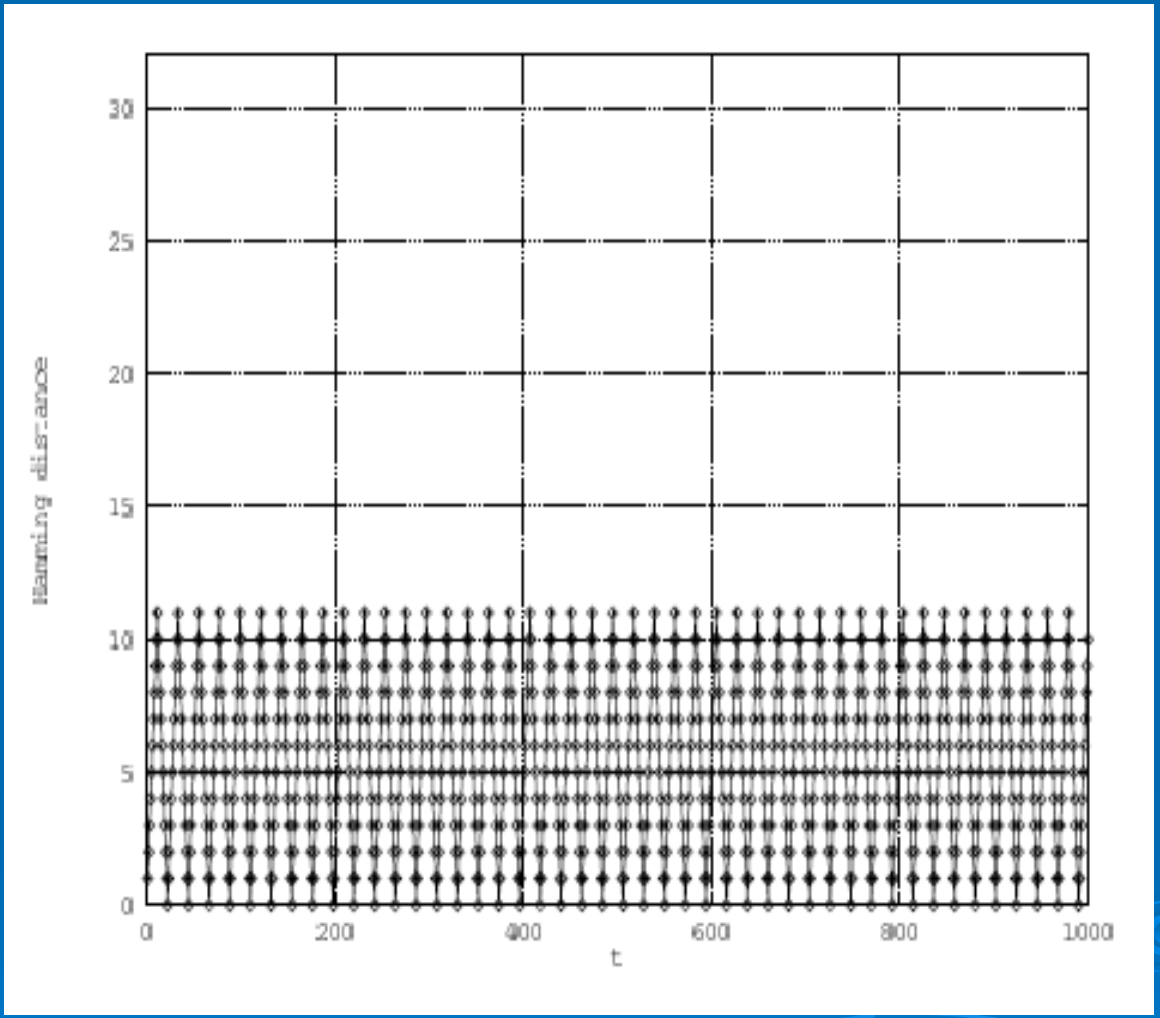


- Curse of dimensionality, “basin filling”

$$L = 64,\ C_5 = 8\ 303\ 633,\ C_4 = 679\ 121.$$

How to escape from an attractor

- Fixed-TS

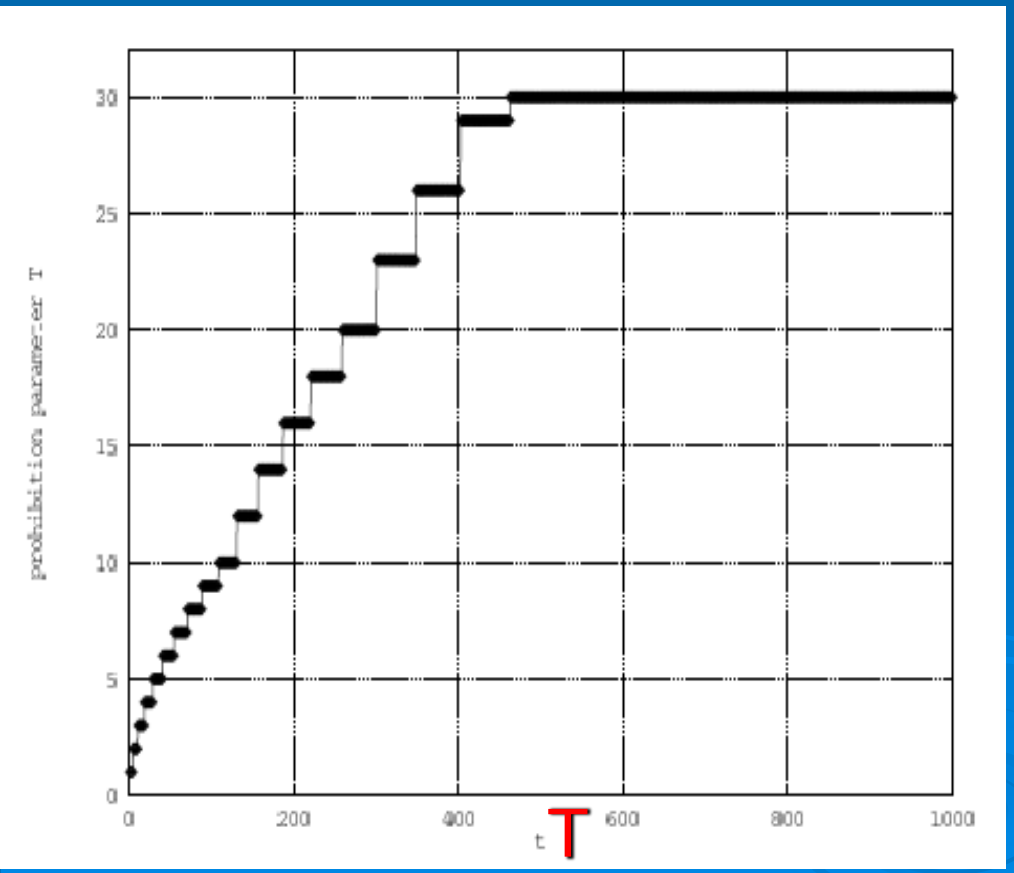


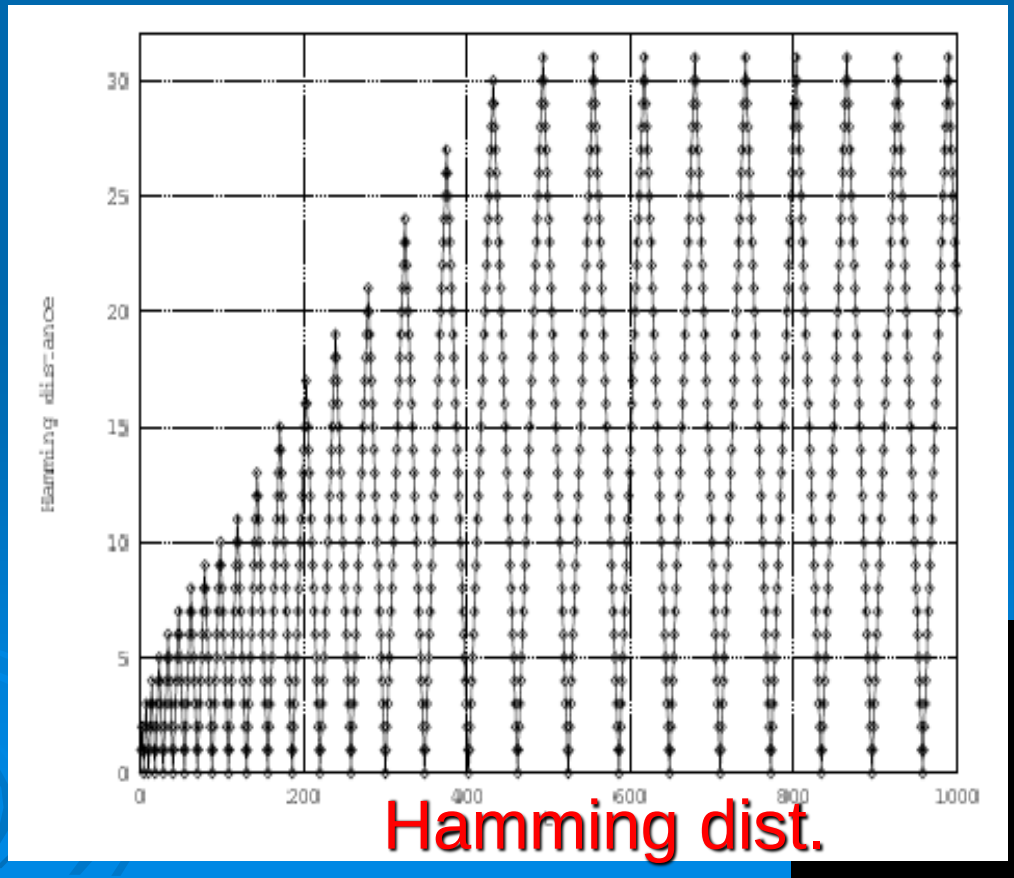
?Sufficient to escape or not ?

How to escape from an attractor

- Reactive-TS (react when loc. minimum is repeated)

$$\textsc{React}(T) \quad = \quad \min\{\max\{T \times 1.1, T+1\}, L-2\}$$





How to escape from an attractor

- Reactive-TS

$$
\begin{aligned}
t(T) &= \sum_{i=1}^{T} 2(i+1) = 3T + T^2 \\
t(H_{max}) &= \left(H_{max}^2 + H_{max} - 2\right) \\
H_{max}(t) &= \frac{1}{2}\left(\sqrt{9+4t} - 1\right)
\end{aligned}
$$

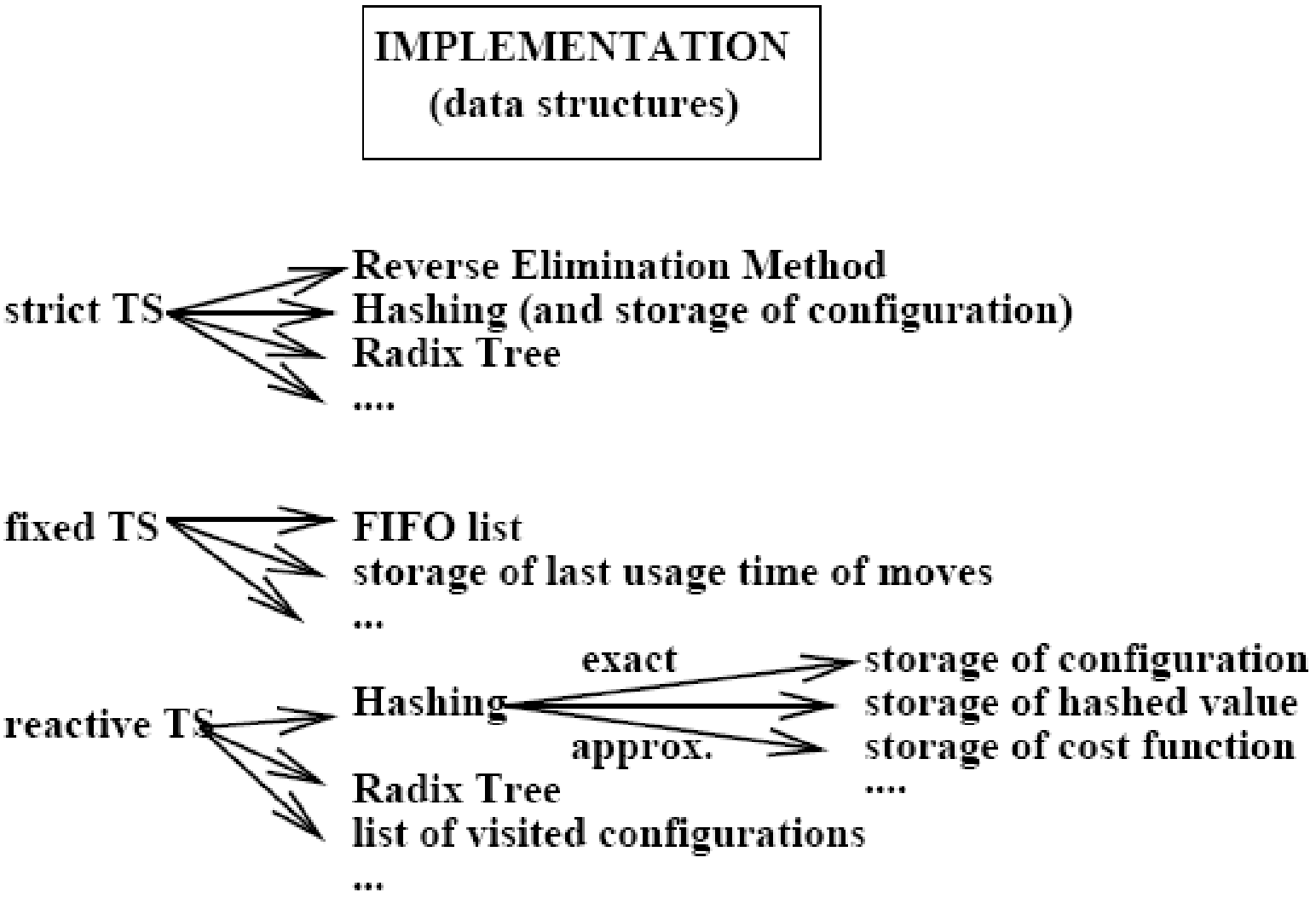
- reachable Hamming distance is approximately O(√t) during the initial steps.
- Qualitative difference: an (optimistic) logarithmic increase in the strict algorithm, and a (pessimistic) increase that behaves like the square root of the number of iterations in the reactive case.

Dynamical systems versus implementation (policies vs mechanisms)

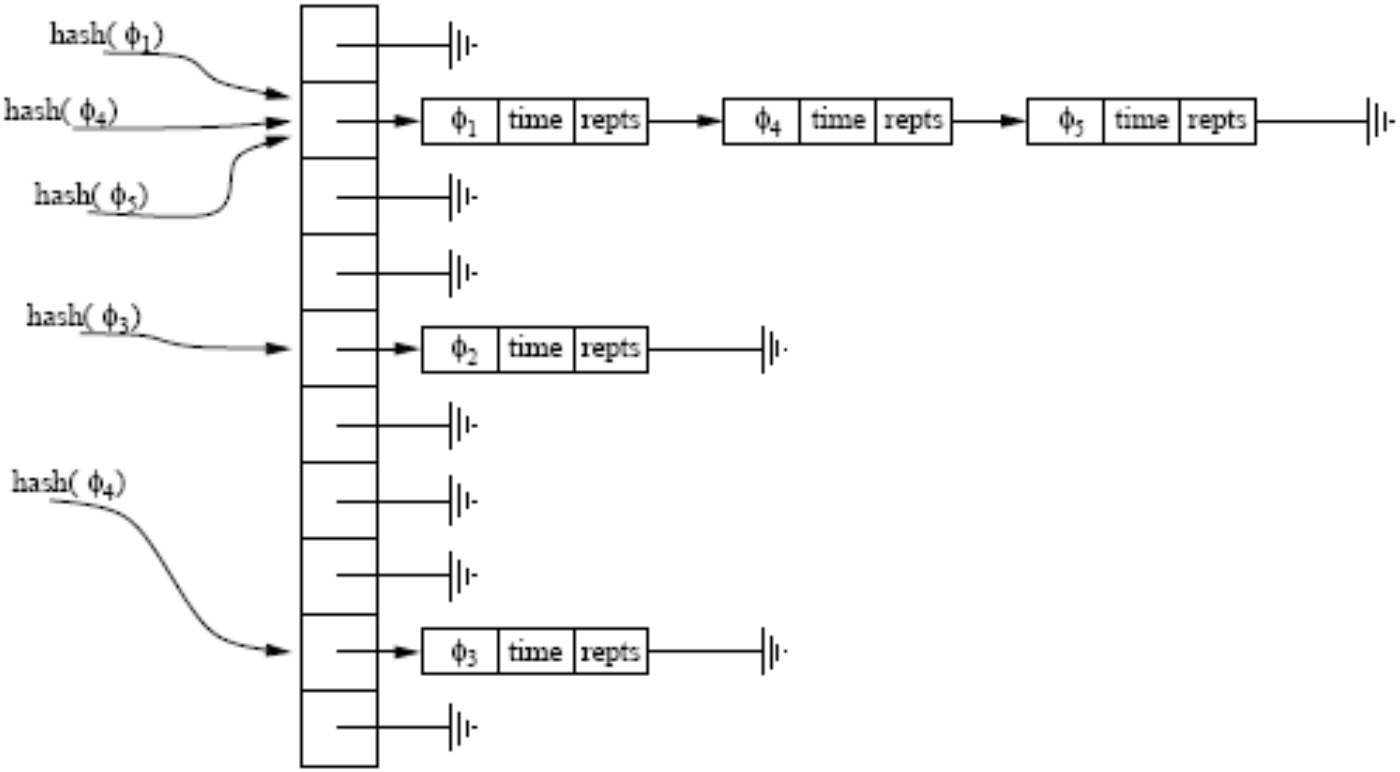
DISCRETE DYNAMICAL SYSTEM
(search trajectory generation)

DETERMINISTIC	STOCHASTIC
* strict TS	* probabilistic TS
* fixed TS	* robust TS
	* fixed TS with stochastic tie breaking
* reactive TS	* reactive TS with stochastic tie breaking * reactive TS with neighborhood sampling (stochastic candidate list strategies)

Dynamical systems versus **implementation**

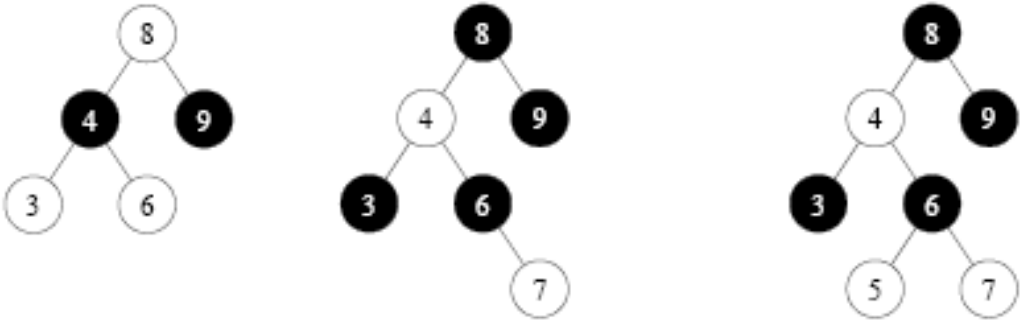


Fingerprinting!



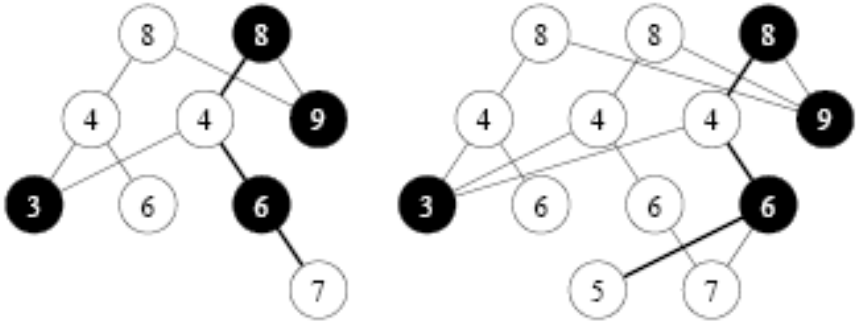
Persistent dynamic sets

(a) Ephemeral red-black tree



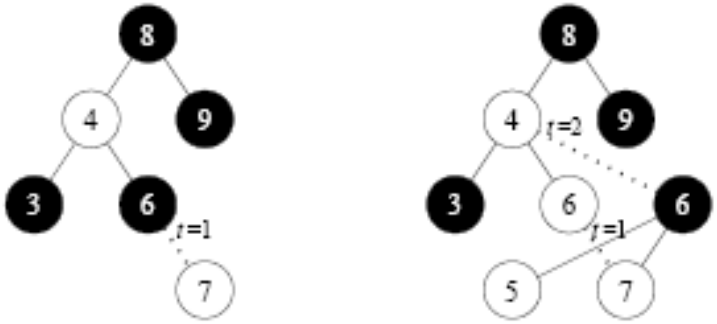
$t = 0$ $t = 1$ $t = 2$

(b) Persistent red-black tree with path copying



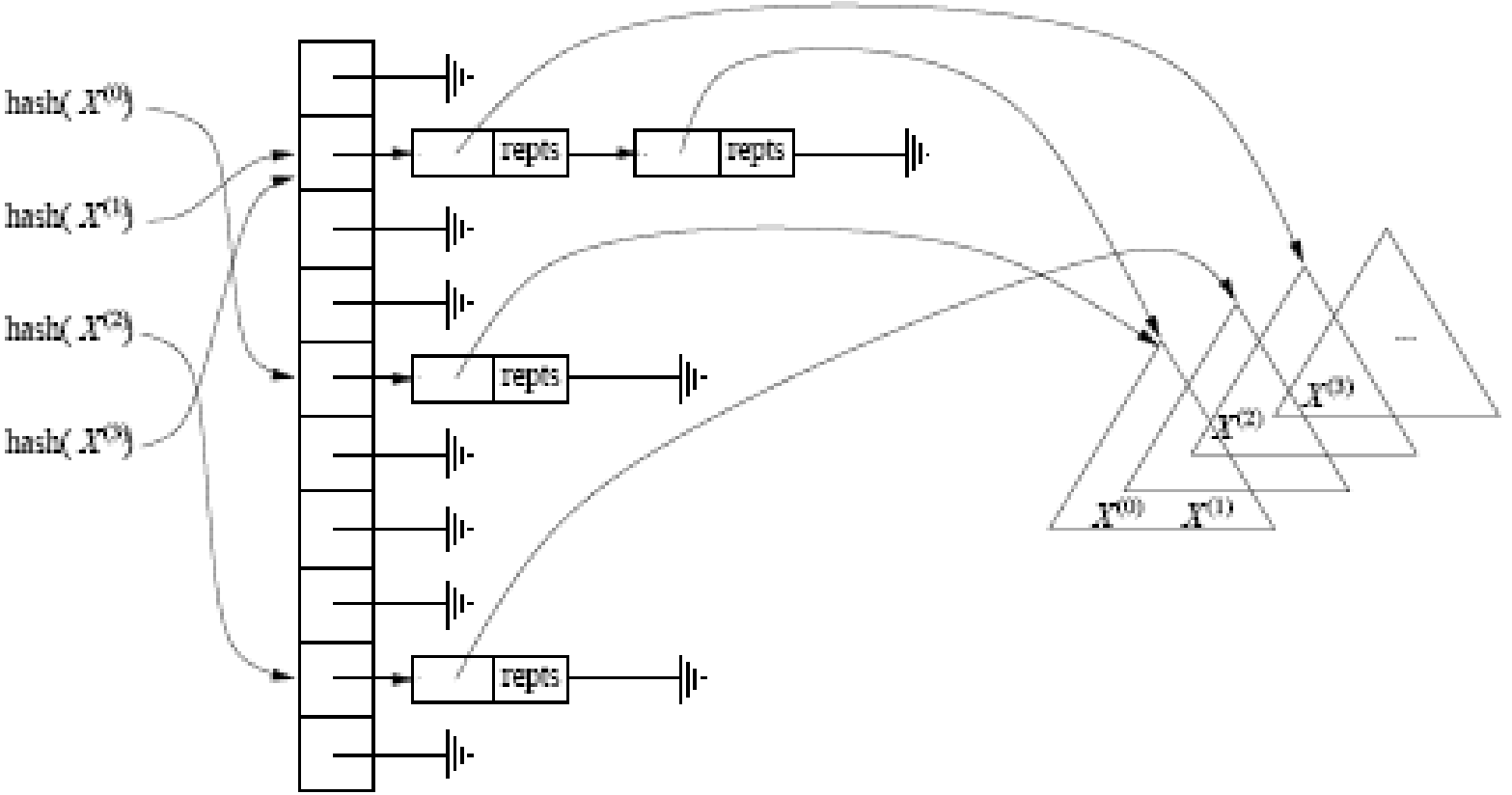
$t = 1$ $t = 2$

(c) Persistent red-black tree with limited node copying



$t = 1$ $t = 2$

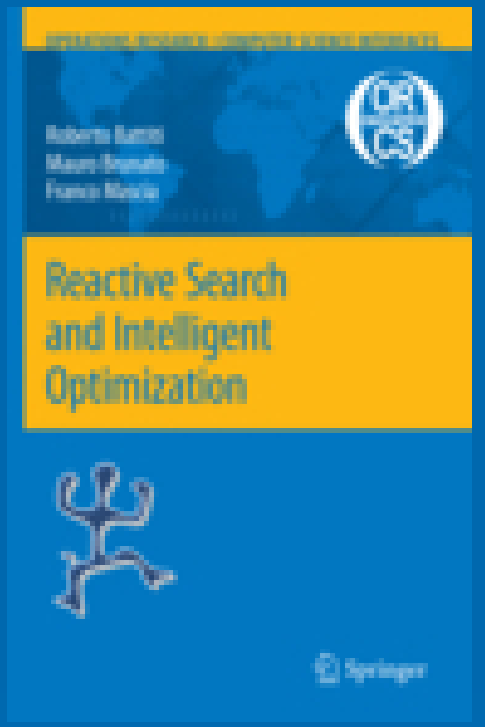
Open hashing with persistent sets



each iteration of *reactive-TS* requires O(L) average-case time and O(1) amortized space for storing and retrieving the past configurations and for establishing prohibitions.

Other reaction opportunities

- **Variable Neighborhood Search**
- **Iterated Local Search**, kicks, **…**
- **Annealing schedule**
- **Objective function modifications**, tunneling, dynamic local search
- **Model-based search**
- **Different randomness levels** (SAT)
- **Algorithm portfolios and restart**
- **Racing**



RSO context

f(x) is given (either analytically or as a black box)

the emphasis is on learning local models of the fitness landscape and using them while optimizing (*no additional knowledge from DM required*)

… but in some cases f(x) to optimize is not given, modeling user preferences is a crucial issue

Try asking a decision maker:
"give me the f(x) that you are optimizing"

Learning *what* to optimize

Example: MOP: Finding a partner: *intelligence* versus *beauty*

How many IQ points for one less beauty point?

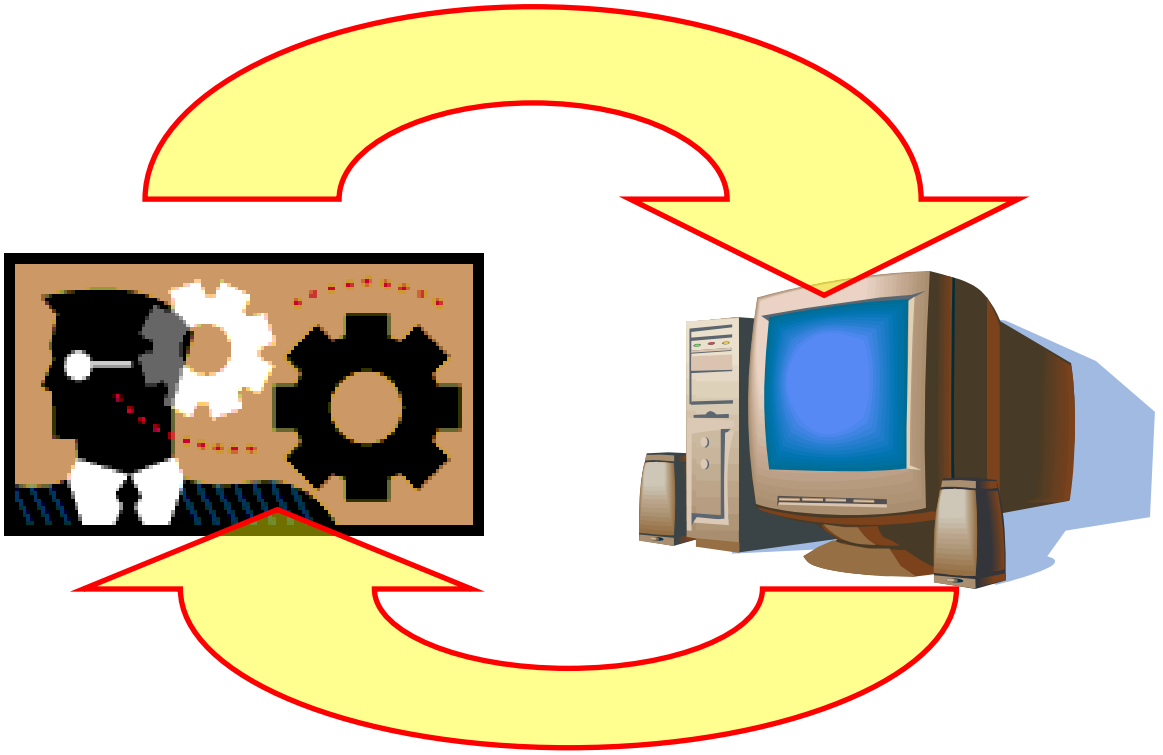
Is beauty more important than intelligence for you? By how much?

Effective optimization as iterative process with learning

Flexible and interactive decision support and problem solving

Crucial decisions depend on factors and priorities which are not always easy to describe before.

Feedback from the user in the exploration phase!



Multiobjective optimization

intermediate (classical) case of missing knowledge:

some criteria are given f1(x) f2(x) … fk(x)

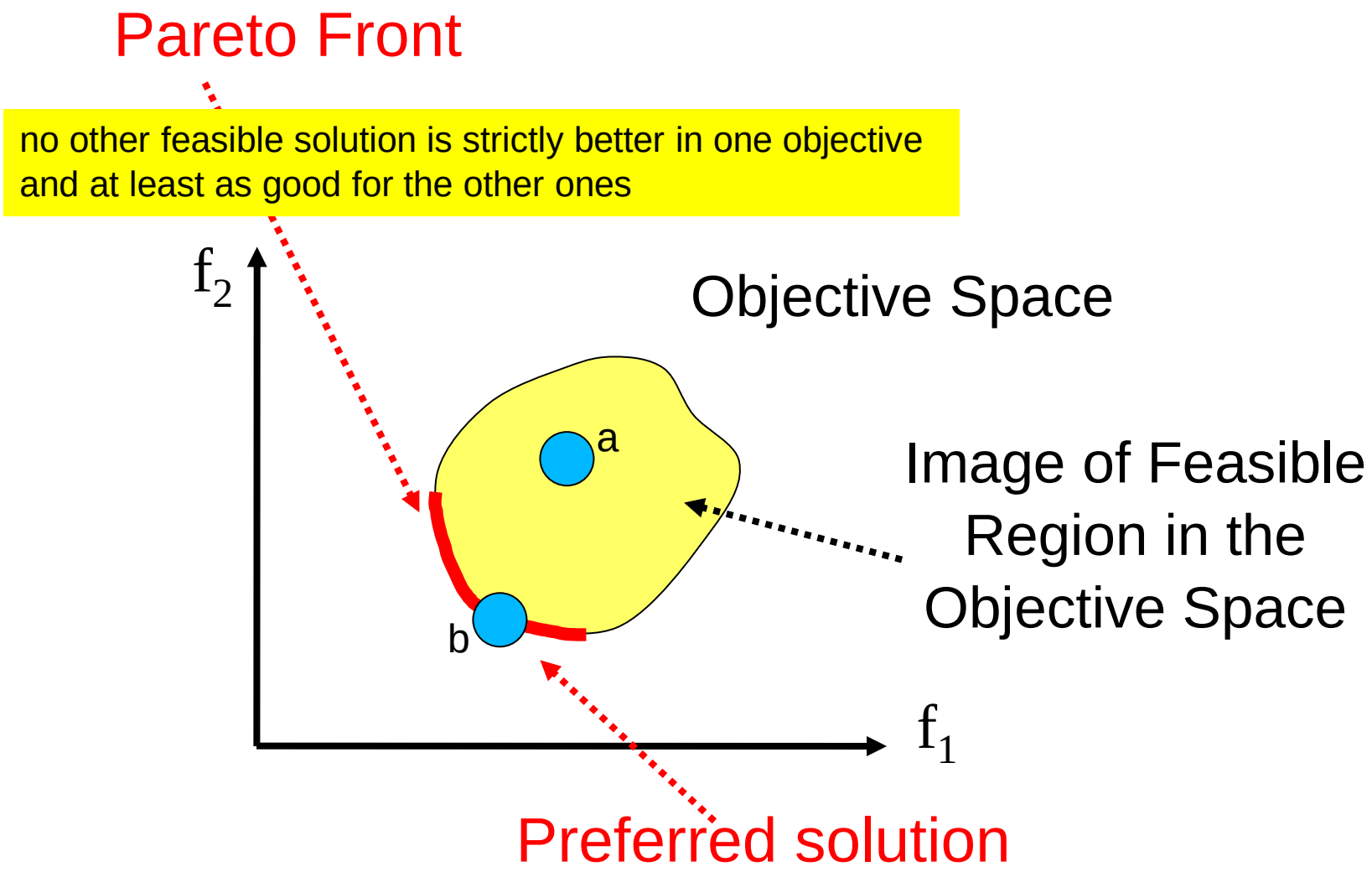
but not easily **combined** into a single f(x)

…provide efficient vector solutions (f1,…,fk)

leave to the user the possibility to *decide*

(and to *learn* about possibilities and “real” objectives, even if not formalized)

Efficient frontier (PF)



Preference information

- Critical task: identify the preferred solution for the DM from the efficient frontier

- Based on the DM preference information usage:
  - *A priori* MOO methods
  - *A posteriori* MOO methods
  - *Interactive* MOO methods (IM)

A priori methods

- Assumptions about preference information before optimization process
- DM specifies preference on the objectives a priori
- Drawbacks:
  - Very difficult task for DM
  - DM often does not know before how realistic his expectations are (no learning possiblities)

A posteriori methods

- The Pareto optimal set (or part of it) is generated and presented to the DM who selects the most preferred among the alternatives.
- Drawbacks:
  - Generation is computationally expensive: find all the non dominated solutions!
  - Hard for the DM to select among a large set of alternatives Decision paralysis
  - Presenting / displaying the alternatives to the DM

Interactive methods

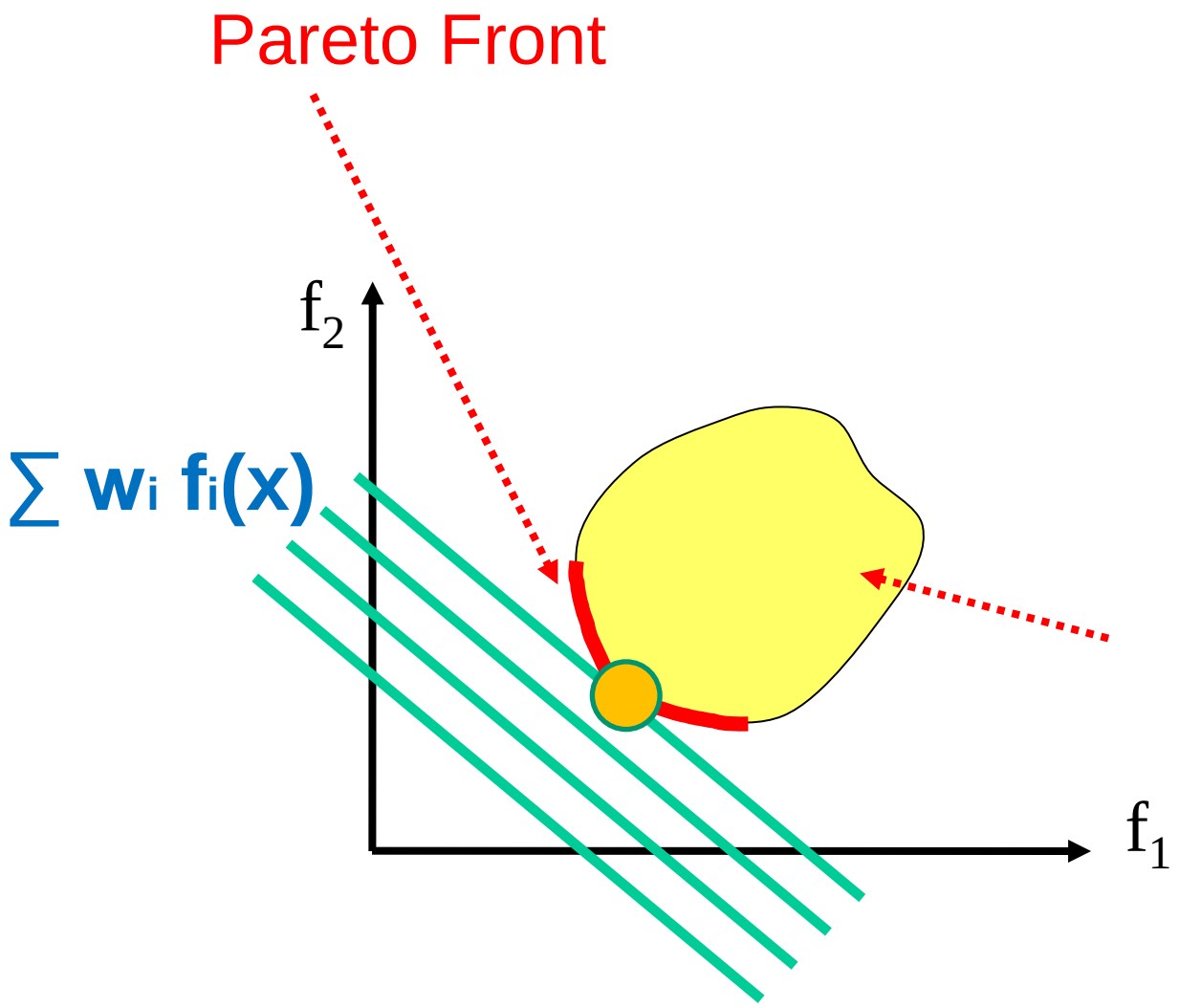
- Solutions generation phases alternated to solution evaluation phases requiring user interaction
- Effective approach
  - Only a subset of the Pareto optimal set has to be generated and evaluated by the DM
  - The DM drives the search process
  - The DM gets to know the problem better (learning on the job)

Interactive methods

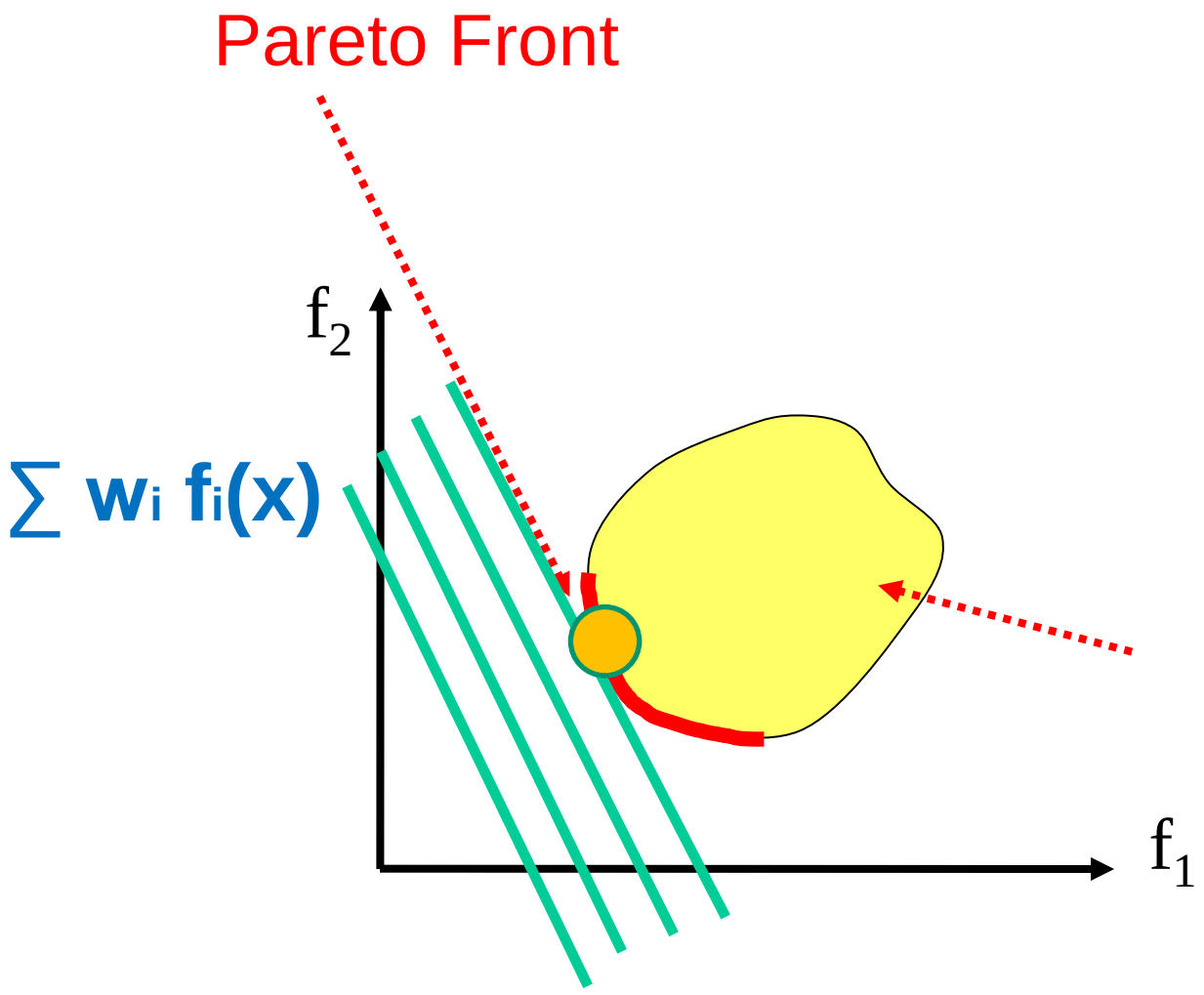
- Choices:
  - How information is provided to DM
  - How preference information is obtained from DM
  - How the search process is updated based on the preference information
  - How the original MOO problem is transformed into a single-objective optimization problem (scalarization process)

    e.g. optimize: $\sum w_i f_i(x)$

Scalarization



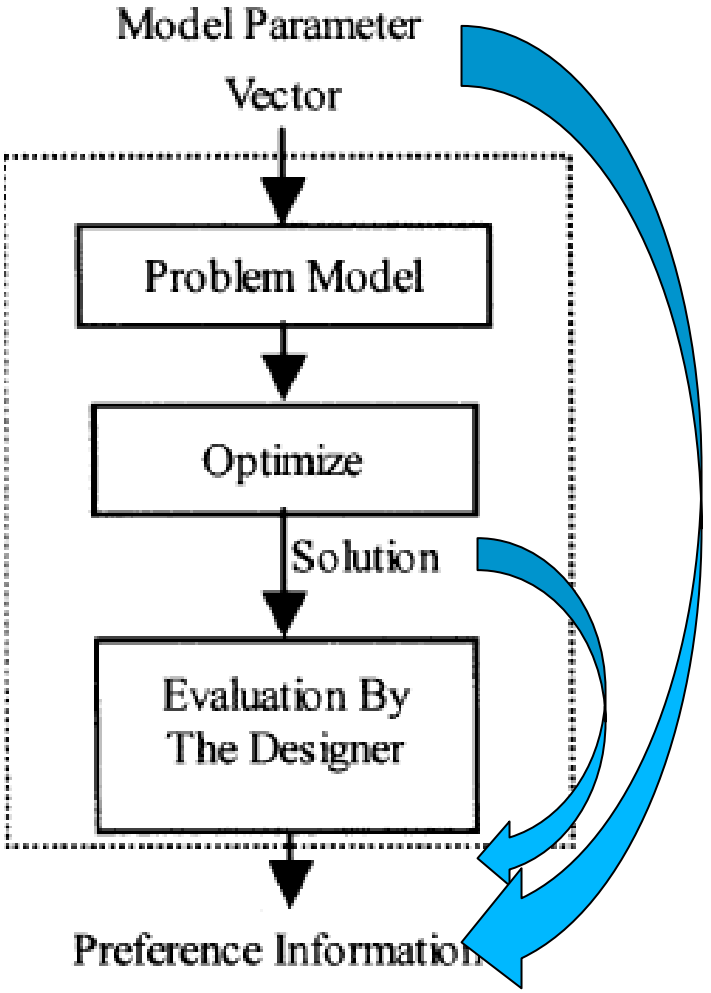
Scalarization



Intelligent Interactive MO Method

(Huang *et al.* 2005)

- Iterative procedure
- NN model of the preference information structure
  - Input: model parameter vector
  - Output: preference value
- At each iteration reduce the model parameter vector space based on DM preference information



Current work: B-C EMO: learning for multiobjective optimization

IEEE Transactions on Evolutionary Computation, Vol. 14, Issue 15, pag. 671 - 687, 2010.

Context is the same: learning user preferences

1. Train a predictor able to reproduce **user preferences**
2. Use the learned predictor to **guide the search** in place of the user

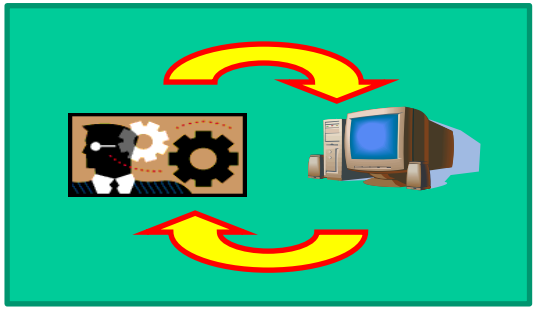
Emphasis is different:

DM time is a scarce and costly resource It is crucial to minimize the number of queries made to the user and their complexity

Robustness for noise (inconsistencies), model flexibility

Population-based approach (EMO)

Brain-Computer EMO



Issues:

- **Bounded-rationality and information bottlenecks** → *satisfacing solutions (Simon)*
- **Learning on the job** – the DM is building a conviction of what is possible and confronting this knowledge with his preferences, that also evolve
- **Simple questions** (comparison, qualitative evaluation,...)
- **Uncertainty and inconsistency**

Formalizing user preferences

$$U(\mathbf{z}) = \sum_{k=1}^{m} w_k z_k .$$

Ideal objective vector

$$U(\mathbf{z}) = -\left(\sum_{k=1}^{m} (w_k |z_k^* - z_k|)^p \right)^{1/p}$$

L∞ metric

augmented weighted Tchebycheff program (AWTP):

utopian objective vector

$$\min_{\mathbf{x},\alpha} \quad \alpha + \rho \sum_{k=1}^{m} (z_k^{**} - z_k) \qquad (4)$$

Properly Pareto-optimal
Improvement possible only with **reasonable** worsening

subject to:

$$w_k(z_k^{**} - z_k) \leq \alpha$$

$$\mathbf{w} \in \mathbb{R}^m,\ w_k \geq 0,\ \sum_{k=1}^{m} w_k = 1$$

$$f_k(\mathbf{x}) = z_k,\ \mathbf{x} \in \Omega$$

$$k = 1, \ldots, m$$

Our solution aims at:

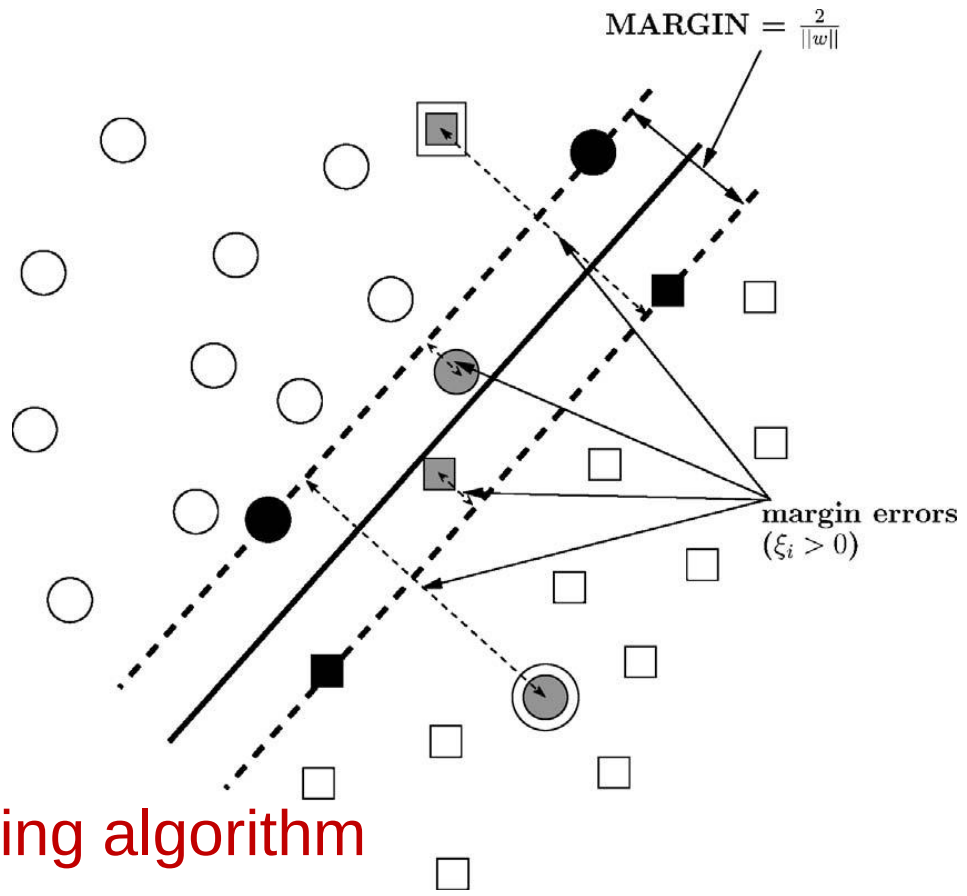
- 1) Learning an arbitrary U from feedback interactively provided by the DM
- 2) Using only DM holistic judgements
- 3) Accounting for incomplete, imprecise and contradictory feedback
- 4) Using directly the learned U to guide the search for refined solutions

SVN: learning to rank (1)

$$\min_{\mathbf{w}\in\mathcal{Z}, b\in\mathbb{R}, \boldsymbol{\xi}\in\mathbb{R}^s} \quad \frac{1}{2}||\mathbf{w}||^2 + C\sum_{i=1}^{s}\xi_i$$

subject to:

$$y_i(\langle \mathbf{w}, \mathbf{z}_i\rangle + b) \geq 1 - \xi_i$$
$$\xi_i \geq 0$$
$$i = 1, \ldots, s.$$



Trading-off **fitting** with
large-margin separation

Using **kernel** to decouple learning algorithm
From example representation

SVN: learning to rank (2)

Lean an utility function for ranking U(**z**) = <**w**,Φ(**z**)>

$$\min_{\mathbf{w}\in\mathcal{Z},\,\xi\in\mathbb{R}^*} \quad \frac{1}{2}\|\mathbf{w}\|^2 + C\sum_{i=1}^{s}\sum_{h_i,k_i}\xi_{i,h_i,k_i} \qquad (8)$$

subject to:

$$\langle \mathbf{w}, \Phi(\mathbf{z}_{h_i}^{(i)})\rangle - \langle \mathbf{w}, \Phi(\mathbf{z}_{k_i}^{(i)})\rangle \geq 1 - \xi_{i,h_i,k_i}$$

$$\xi_{i,h_i,k_i} \geq 0$$

$$h_i, k_i : y_{h_i}^{(i)} < y_{k_i}^{(i)}$$

$$i = 1, \ldots, s.$$

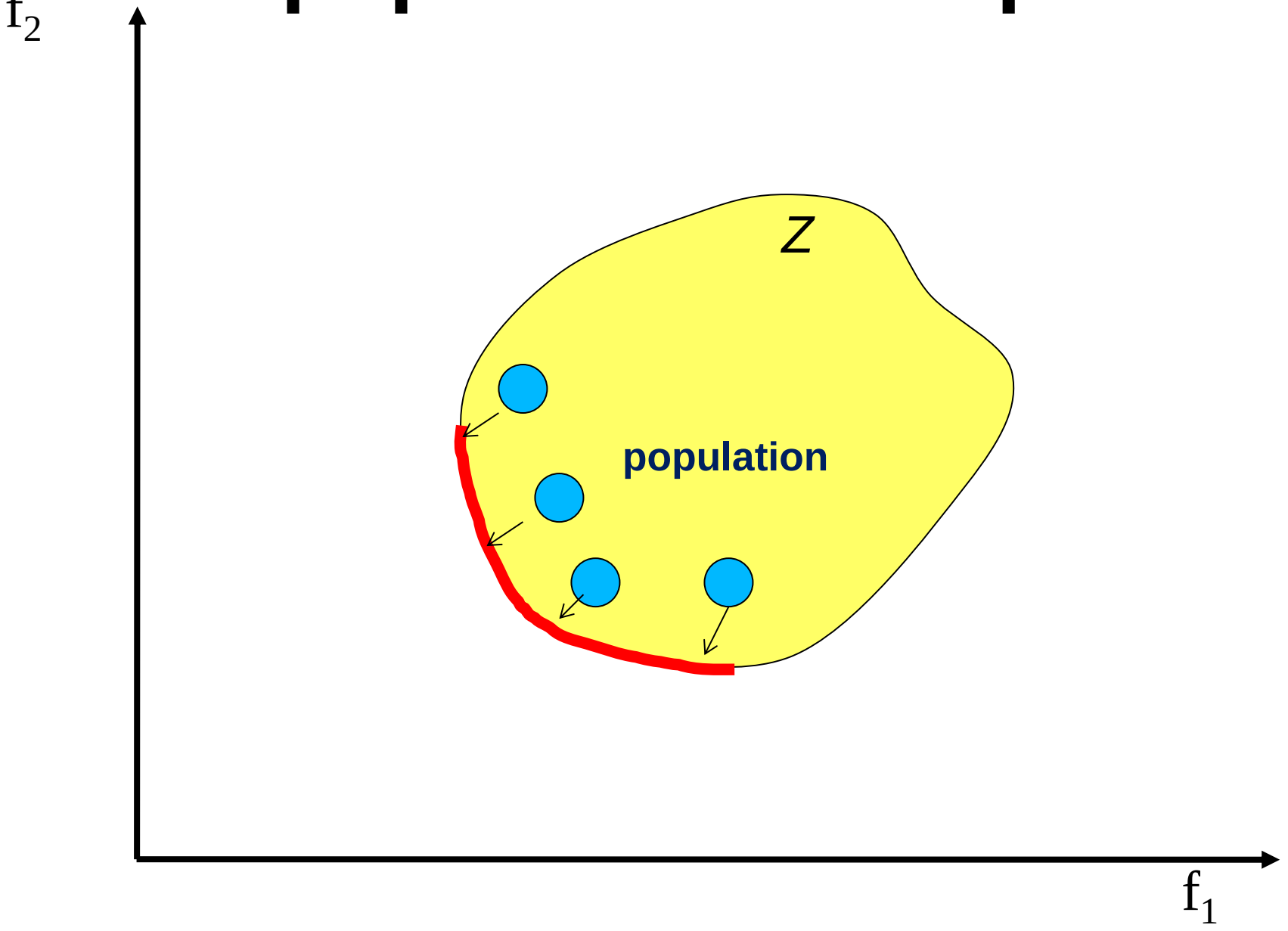
$$U(\mathbf{z}) = \sum_{\mathbf{z}_j^{(i)}\in SV^*} \alpha_{i,j} K(\mathbf{z}_j^{(i)}, \mathbf{z})$$

where

$$\alpha_{i,j} = \sum_{\substack{(\mathbf{z}_{h_i}^{(i)}, \mathbf{z}_{k_i}^{(i)}) \in SV \\ : \ \mathbf{z}_j^{(i)} = \mathbf{z}_{h_i}^{(i)}}} \alpha_{i,h_i,k_i} - \sum_{\substack{(\mathbf{z}_{h_i}^{(i)}, \mathbf{z}_{k_i}^{(i)}) \in SV \\ : \ \mathbf{z}_j^{(i)} = \mathbf{z}_{k_i}^{(i)}}} \alpha_{i,h_i,k_i}$$

For a polynomial $K(\mathbf{z}, \mathbf{z}') = (1 + \langle \mathbf{z}, \mathbf{z}'\rangle)^m$

EMO: a population to map the PF



Combining EMO with ranking SVN

- General scheme usable for every EMO (Evolutionary Multi-objective Optimization, *usually intended to map entire PF*)
- Experimental tests on non-dominated sorting genetic algorithm version II (NSGA-II) EMOA

K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan, “A fast elitist multiobjective genetic algorithm: NSGA-II,” *IEEE Trans. Evol. Computat.,* vol. 6, no. 2, pp. 182–197, Apr. 2002.

Evaluating a population

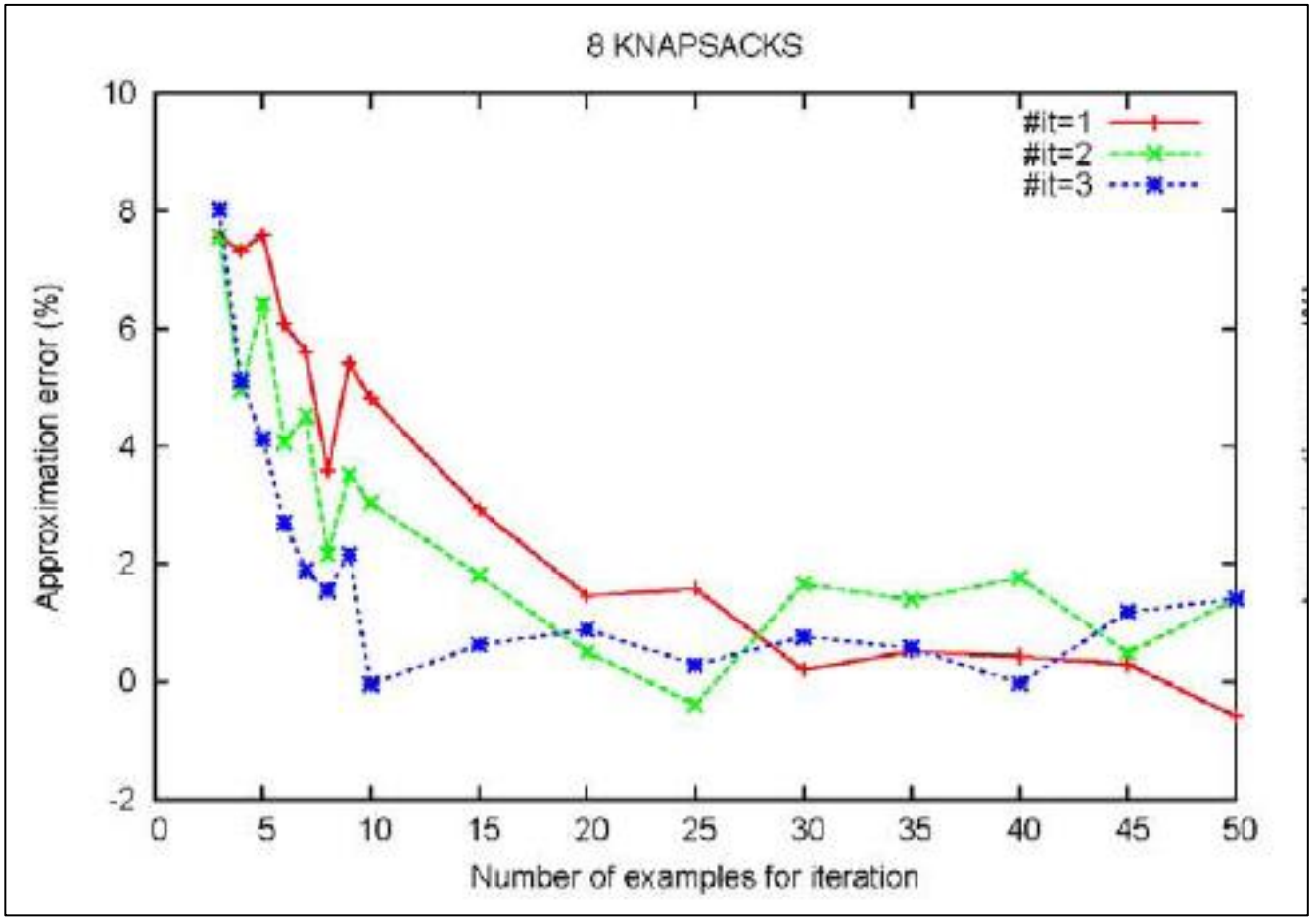
- Unordered population combining parents and offsprings, then
- 1) collecting the subset of non-dominated individuals
- 2) **sorting them according to the learned utility function**;
- 3) appending to the sorted set, the result of repeating the procedure on the remaining dominated individuals.

Training procedure (1)

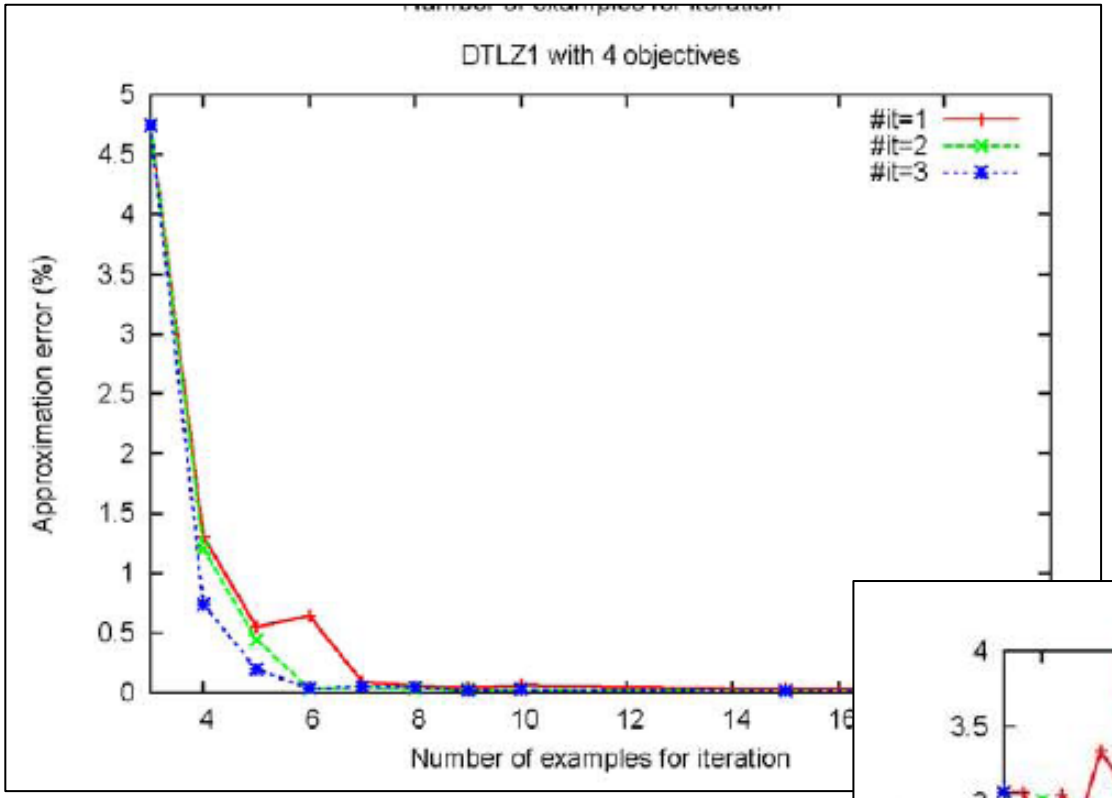
- 1) selecting a set of *exa best training individuals* according to the current criterion (random nondominated individuals at the first iter.)
- 2) collecting pairwise preferences from the DM and adding them to the training set
- 3) performing a kernel selection phase by a *k-fold cross-validation*
- 4) training the utility function on the overall set of examples with he chosen kernel.

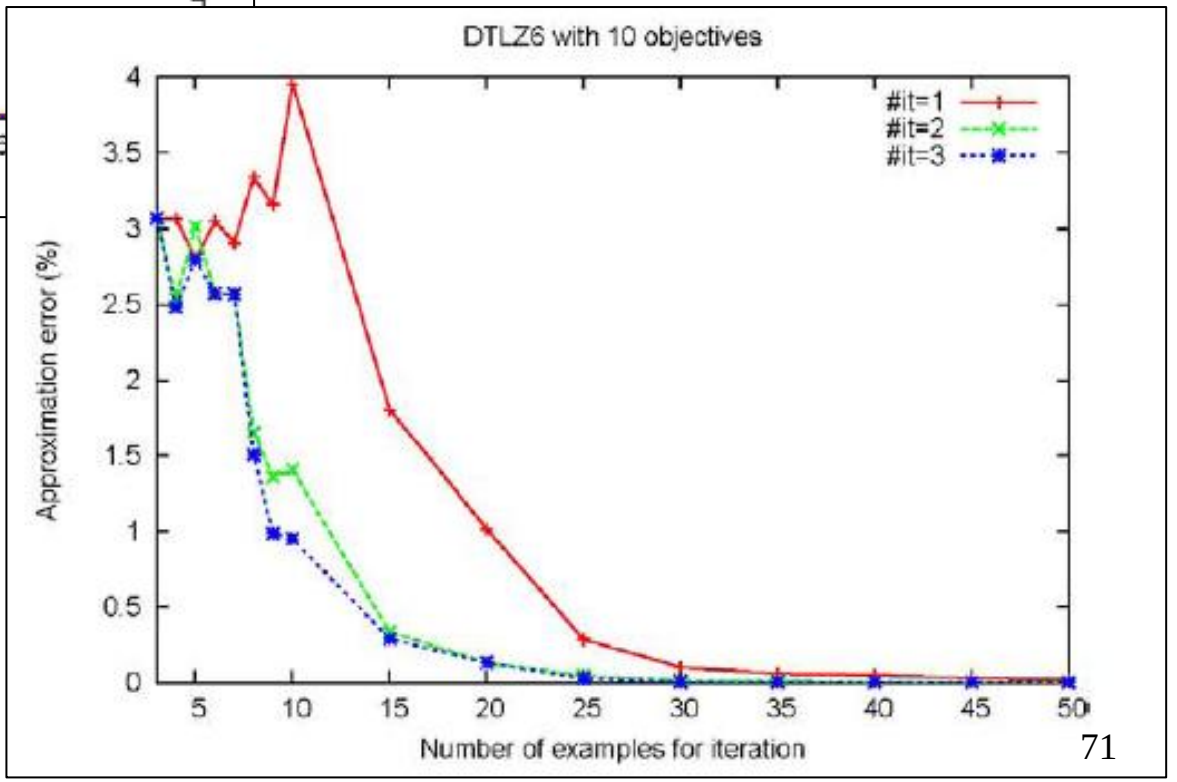
Experiments

- Effectiveness in ealy **focusing on the correct search area with few queries** to the DM



Experiments (3)





Conclusions

LION is about Machine Learning for Optimization

- **Learn f(x)** from data (experiments, decision maker input)
- **Self-tune** parametric (flexible) heuristics to make them more effective on problems (**offline**) but also on individual instances and local characteristics (**online – reactive**)
- More automation → wider adoption of optimization for businesses, more autonomy of the final user

Interesting area for young and open-minded researchers, challenging problems still ahead!

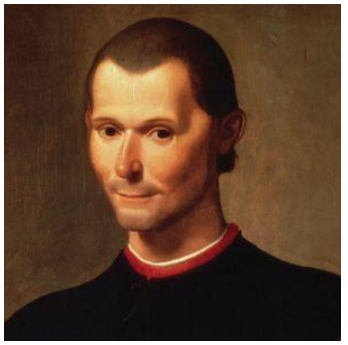
LION11
Submission
Dec 10

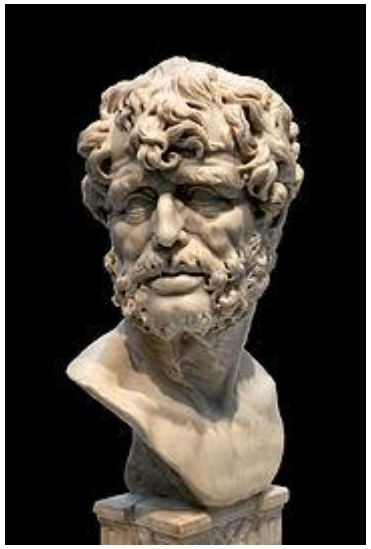
http://intelligent-optimization.org/lion11/

Thank you

Act like the **clever archers (***arcieri prudenti)* who, designing to hit the mark which yet appears too far distant, and knowing the limits to which the strength of their bow attains, **take aim much higher than the mark**, not to reach by their strength or arrow to so great a height, but to be able with the aid of so high an aim to **hit the mark they wish to reach.**

Niccolò Machiavelli , The Prince, c.a. 1500





If one does not know to **which port** one is sailing, **no wind is favorable**.

Seneca, c.a. 50