ALMA MATER STUDIORUM
UNIVERSITÀ DI BOLOGNA

Merging Optimization and Machine Learning

Empirical Model Learning

Michela Milano

ELLIT Workshop: Hybrid AI - 31 Oct 2022

Joint work with…..

Michele Lombardi 

Andrea Bartolini 

Luca Benini 

Andrea Borghesi 

Alessio Bonfietti 

Tias Guns 

Pascal van Hentenryck 

Allegra De Filippo 

Context

- ML and optimization: two way integration
  - In this talk…. mainly combinatorial optimization perspective

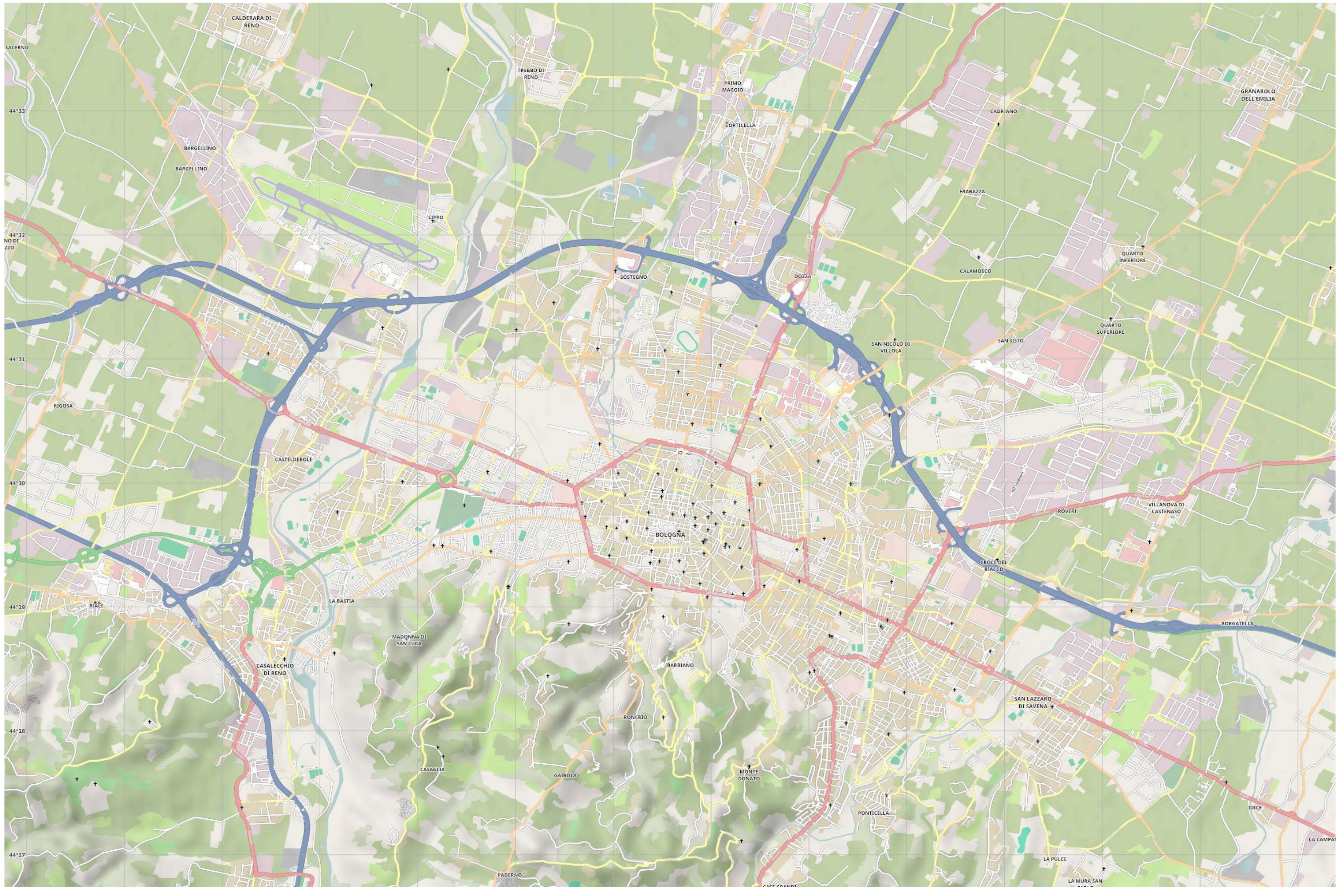
- Many optimization approaches for hosting ML components
  - In this talk…. mainly constraint programming

Motivation

What makes a decision/optimization problem complex?

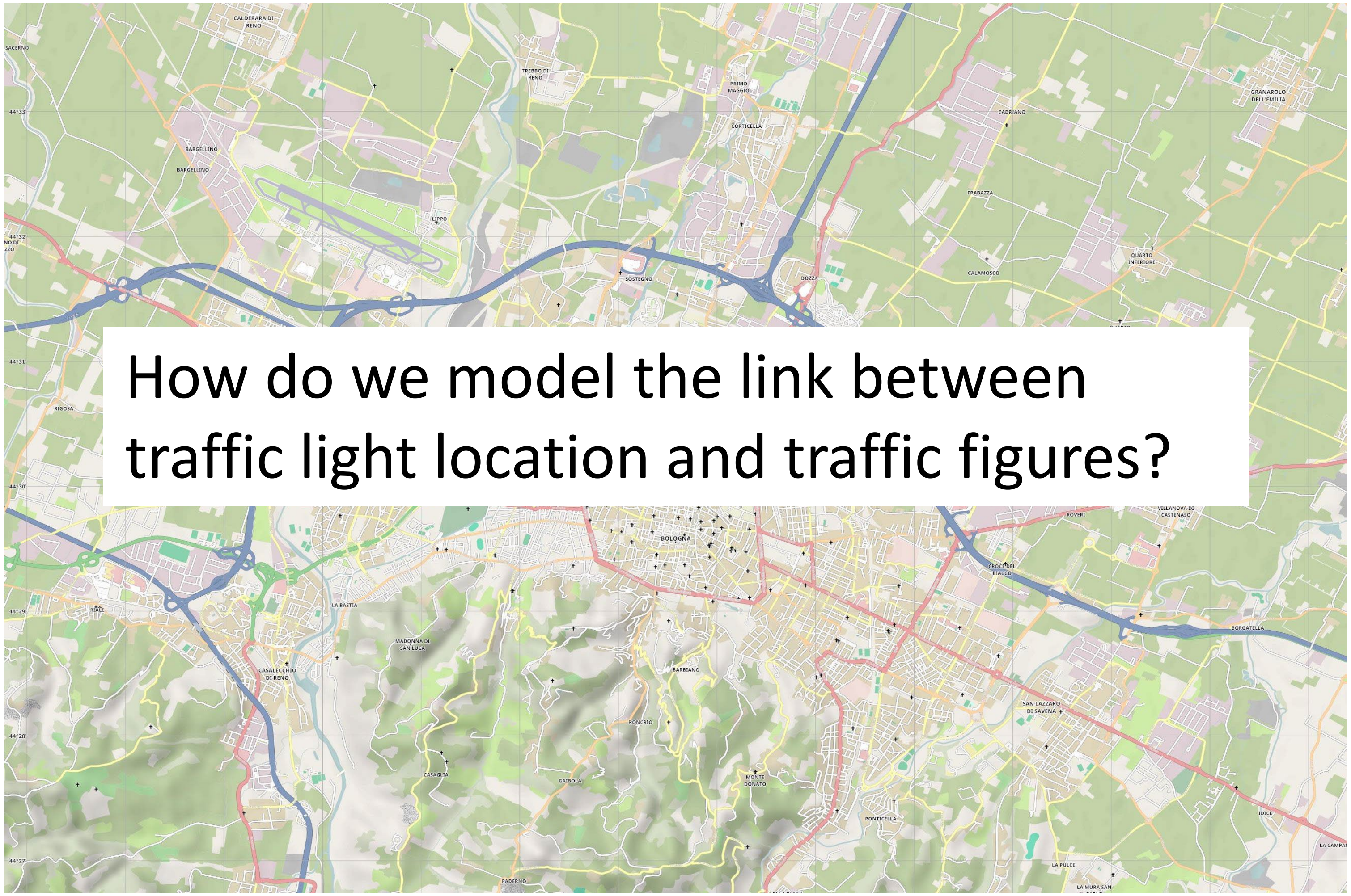
A Case Study: Traffic Light Placement

- Add/remove traffic lights in a city
- Traffic lights can be connected (green wave)
- Every operation has a cost
- Budget limit
- **Objective:** improve traffic flow



A Case Study: Traffic Light Placement

- Add/remove traffic lights in a city
- Traffic lights can be connected (green wave)
- Every operation has a cost
- Budget limit
- **Objective:** improve traffic flow



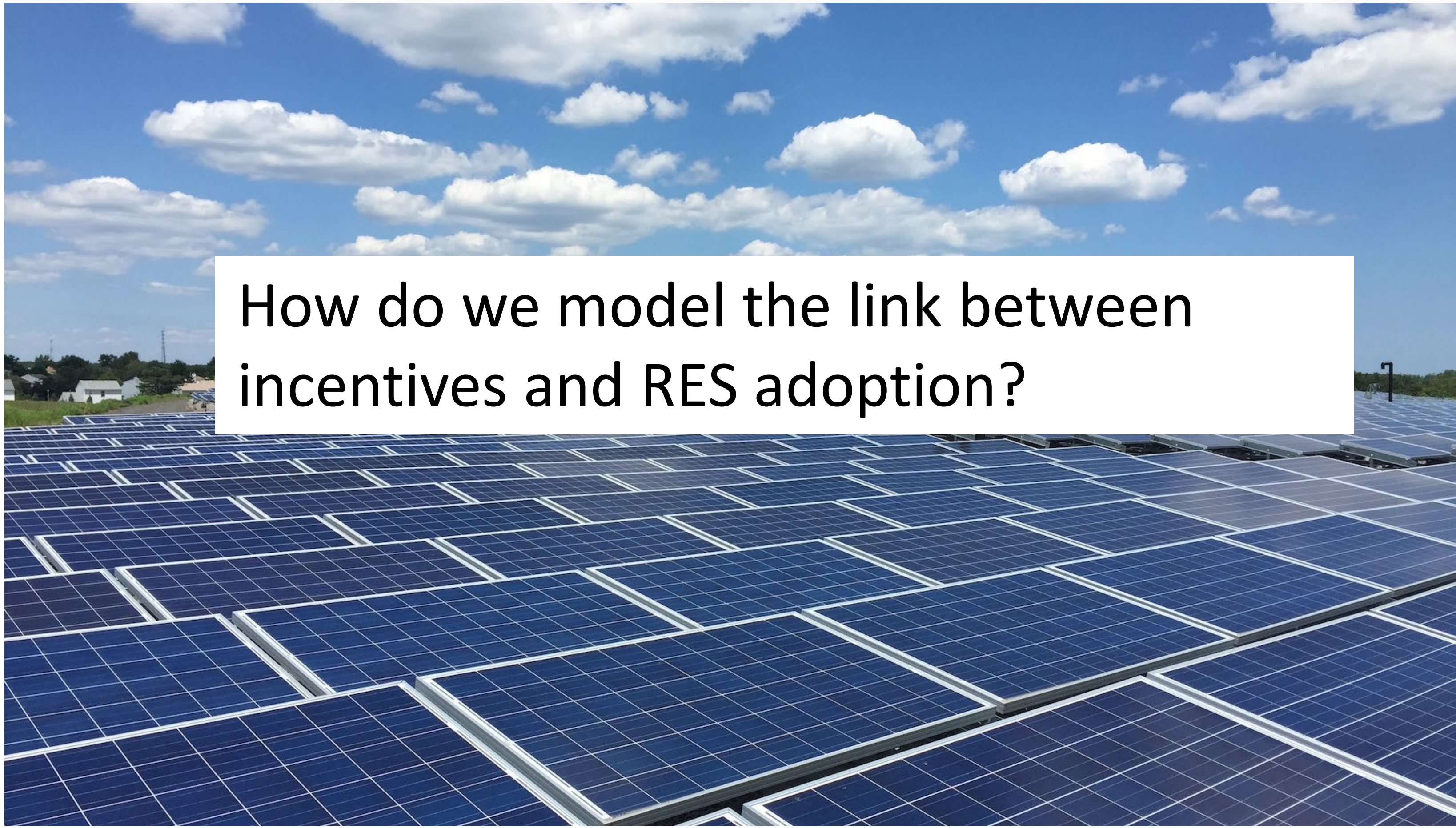
How do we model the link between traffic light location and traffic figures?

A Case Study: RES Incentive Design

- Assign resources to incentive actions
- Reach a renewable generation quota
- **Objective**: minimize cost

A Case Study: RES Incentive Design

- Assign resources to <u>incentive actions</u>
- Reach a renewable generation quota
- **Objective**: minimize cost

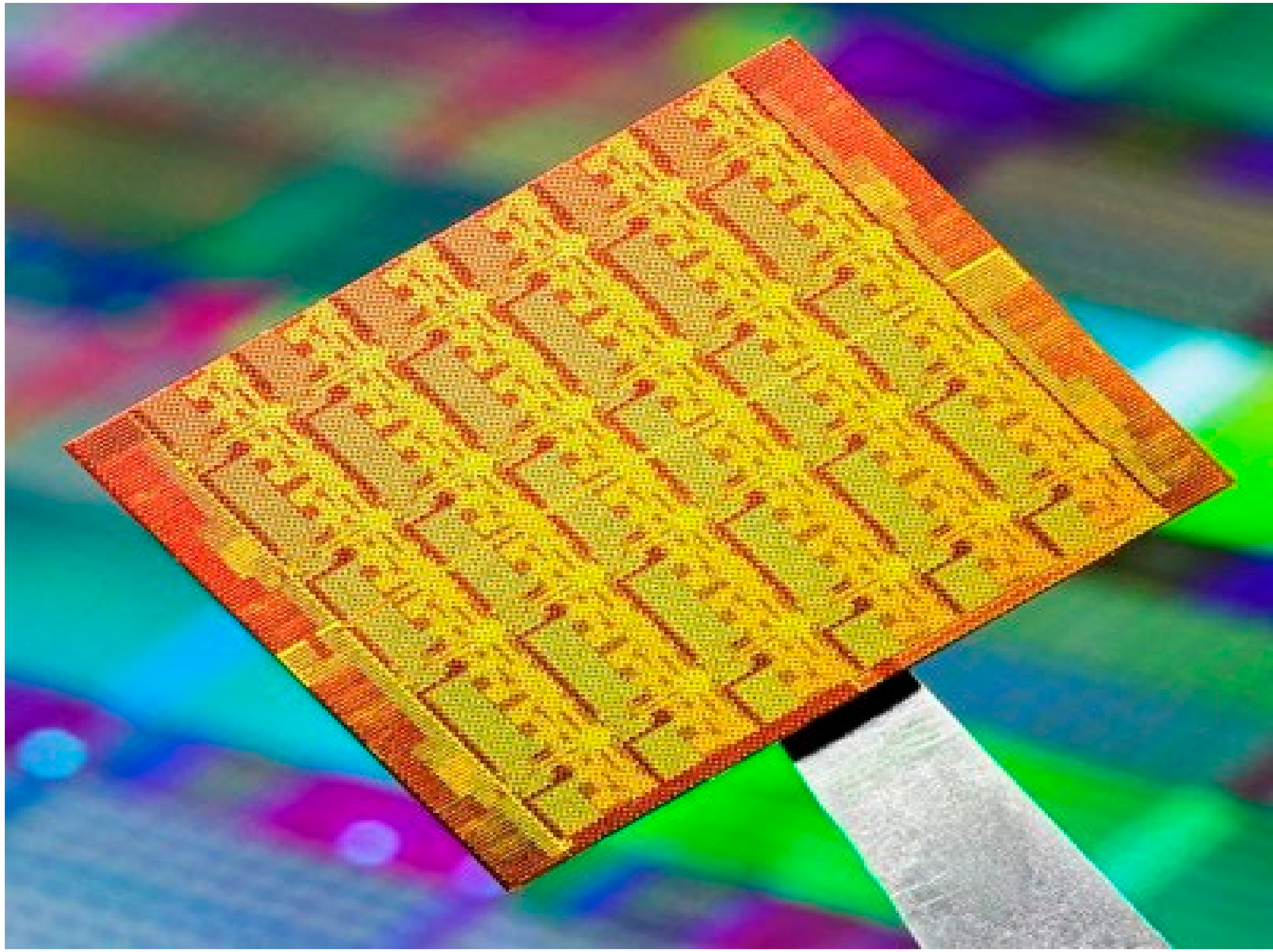


How do we model the link between incentives and RES adoption?

A Case Study: Thermal Aware workload dispatching

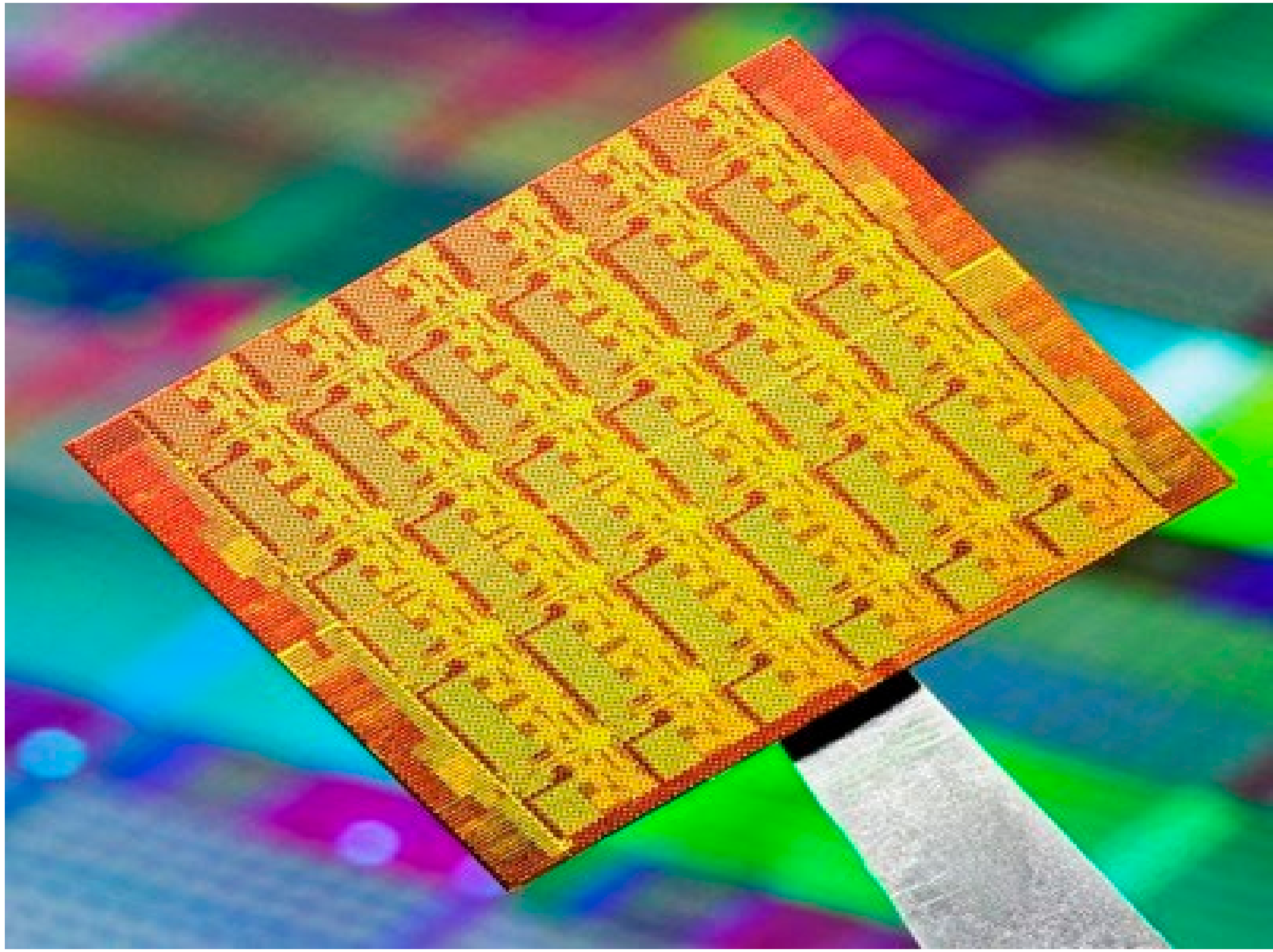
ALMA MATER STUDIORUM UNIVERSITÀ DI BOLOGNA

- Assign software functions to hardware resources (cores/mem)
- Satisfy temporal and QoS constraints
- Keep avg temperature below threshold
- **Objective**: avoid hot spots



A Case Study: Thermal Aware workload dispatching

- Assign software functions (jobs) to hardware resources (cores/mem)
- Satisfy temporal and QoS constraints
- Keep avg temperature below threshold
- **Objective**: avoid hot spots



Temperature is linked to **core efficiency** as there are thermal controllers on the platform

A Case Study: Thermal Aware workload dispatching

ALMA MATER STUDIORUM
UNIVERSITÀ DI BOLOGNA

- Assign software functions (jobs) to hardware resources (cores/mem)
- Satisfy temporal and QoS constraints
- Keep avg temperature below threshold
- **Objective**: avoid hot spots

Temperature is linked to **core efficiency** as there are thermal controllers on the platform

How do we model the link between job dispatching and temperature/efficiency?

Thermal simulator available !!!

What Makes a Problem Complex

In general, many things:

- Scale
- Different types of decisions
- Poor bounds/propagation…

...But for these problems, it's mostly a modeling issue

How do we model:

- The link between traffic light location and traffic figures?
- Between incentives and renewables adoption?
- Between job placement and temperature/efficiency?

Empirical Model Learning is an attempt to address this difficulties

Motivation

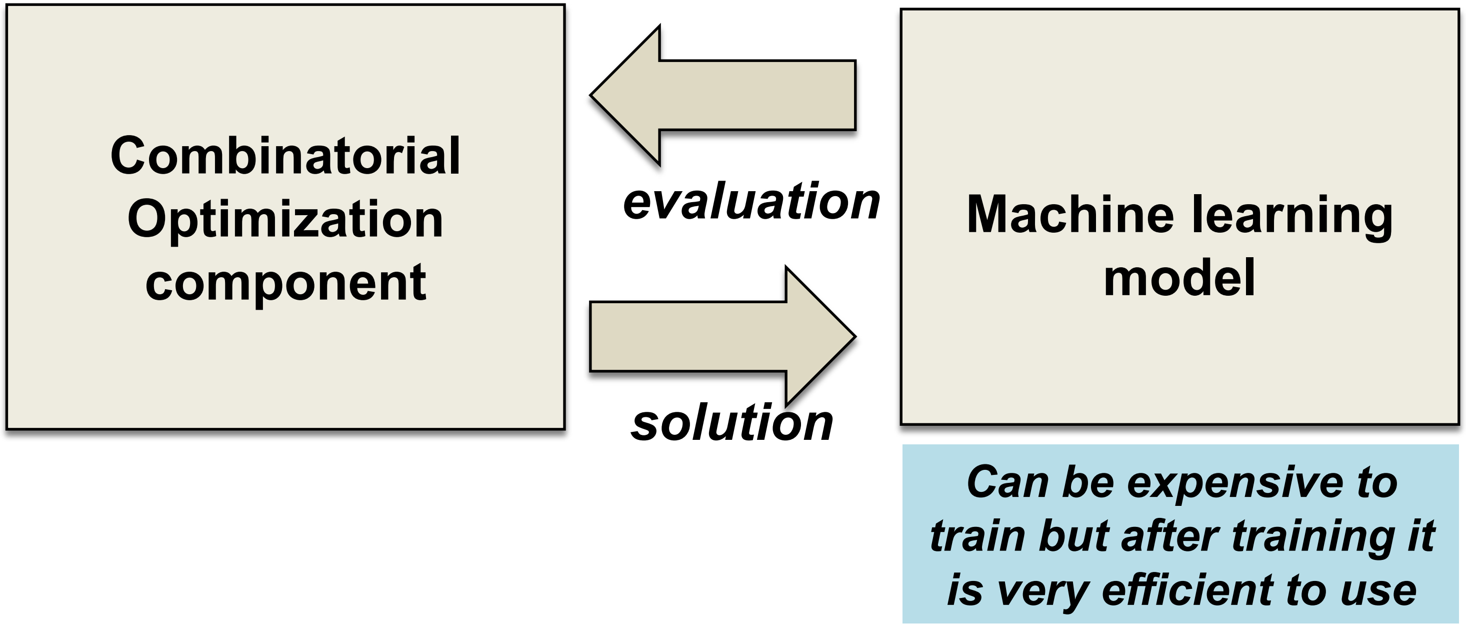
A possible solution:

Use Machine Learning to get a model

Let it **interact in some way** with the optimization component

Light integration

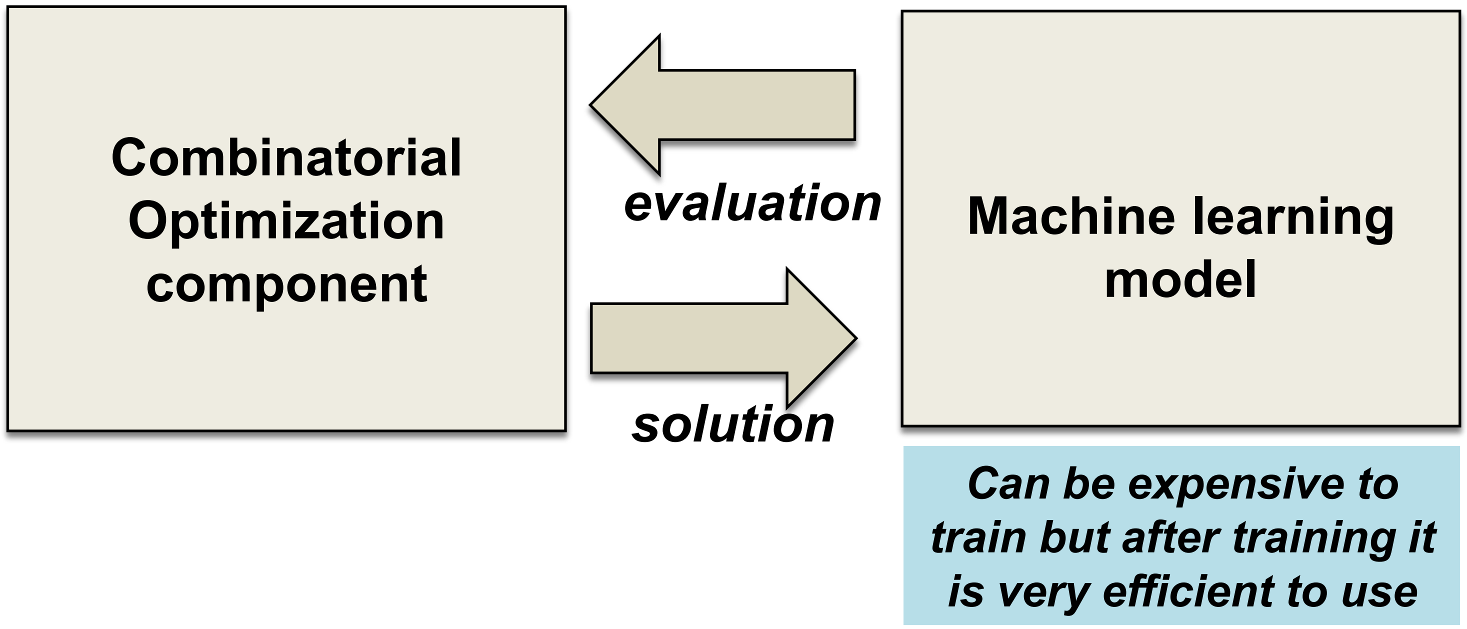
we can have a combinatorial problem solver
passing the solution to a ML model that evaluates it



Surrogate models: [Henao, Maravelias, AIChE 2011], [Cozad, Sahinidis, Miller, 2014]

Light integration

we can have a combinatorial problem solver
passing the solution to a ML model that evaluates it



Generate and test mechanism

Alternative

A possible solution:

Use Machine Learning to get a model

Embed the learnt model into an optimization approach

This is Empirical (Decision) Model Learning

Tight integration

Empirical Model Learning – EML

- The ML model is embedded into the optimization component
- It actively reduces the search space during execution

Tight integration

Empirical Model Learning – EML

- The ML model is embedded into the optimization component
- It actively reduces the search space during execution

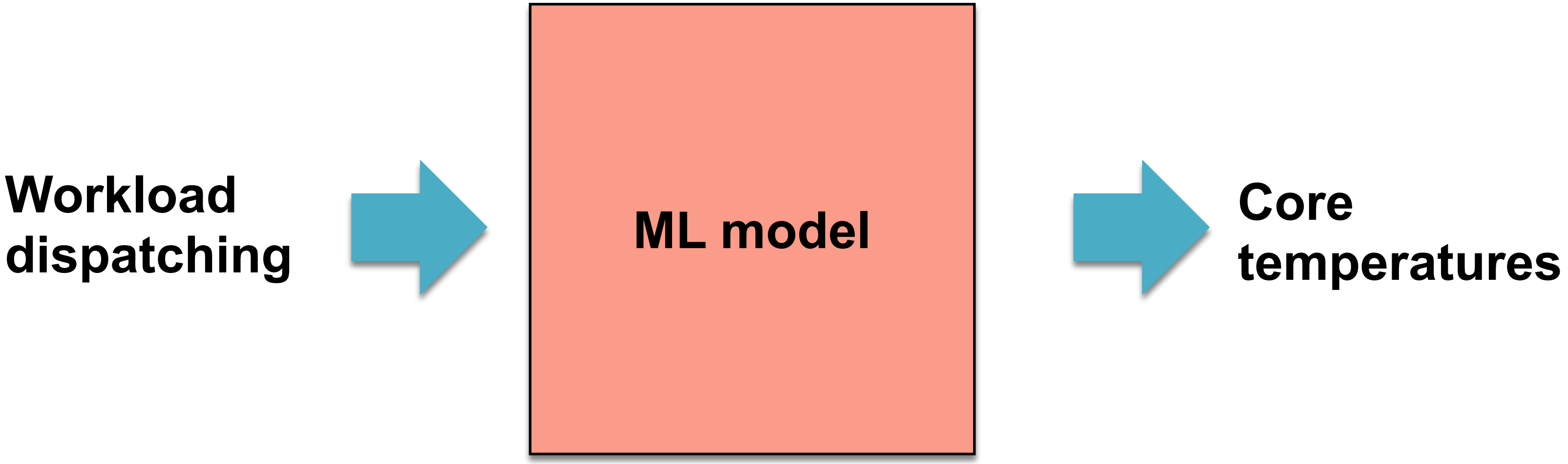
Not limited to objective functions, but also constraints and any relation between decisions and observables

Lombardi, Milano, Bartolini, Empirical Decision Model Learning, AIJ (244), 2017

Tight integration

What is the difference between EML and the traditional use of ML models?

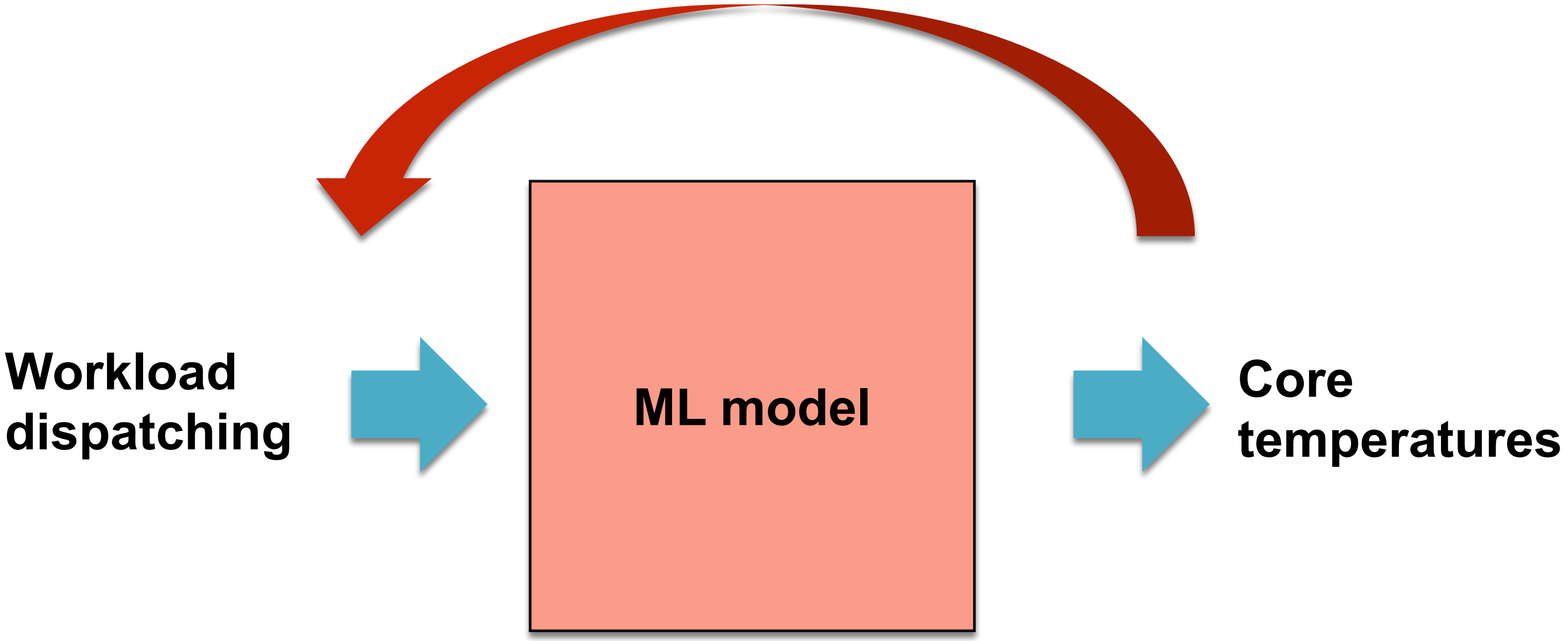
Traditional approaches work forward



Tight integration

What is the difference between EML and the traditional use of ML models?

EML works forward and backward



The ML model becomes a constraint: given temperature limits we remove combinations of workload decisions that lead to inconsistent temperatures

In practice……

Empirical (Decision) Model Learning

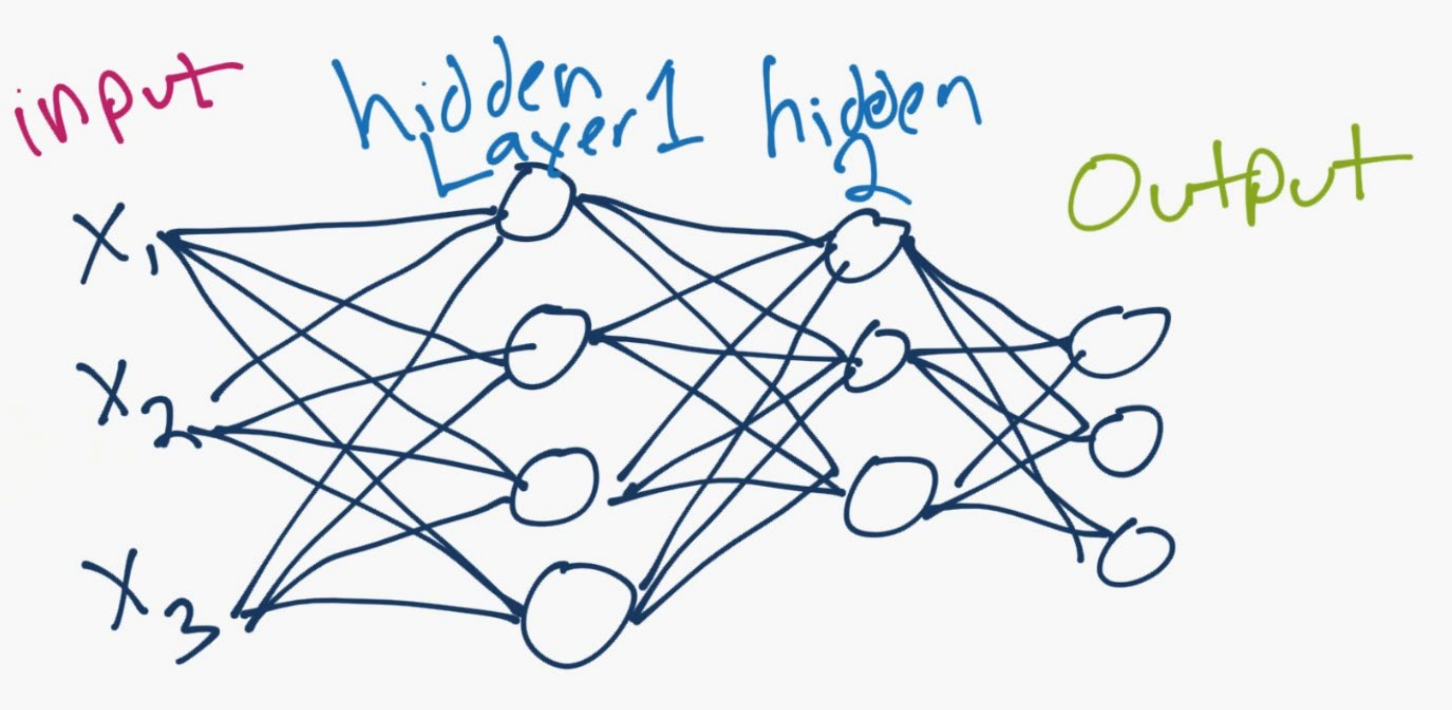
- Start from observations

Avg. Load 0	Std. Load 0	Avg. Load 1	Std. Load 1	…
0.9	0.1	0.7	0.3	…
0.8	0.2	0.8	0.1	…
0.5	0.4	0.6	0.2	…
…	…	…	…	…

Core 0	Core 1	Core 2	…
0.9	0.7	0.8	…
0.7	0.9	0.9	…
0.8	0.6	0.8	…
…	…	…	…

Empirical (Decision) Model Learning

- Start from observations
- Use Machine Learning to get an approximate model



$$h : \text{load stats} \mapsto \text{core } k \text{ eff.}$$

Empirical (Decision) Model Learning

- Start from observations
- Use Machine Learning to get an approximate model
- Embed this "empirical model" in a declarative optimization model

$$
\begin{aligned}
\min z &= f(\vec{x}, \vec{y}) \\
\text{s.t. } \vec{y} &= h(\vec{x}) \\
&\text{all manner of constraints}
\end{aligned}
$$

- x = ML model input
- y = ML model output

Empirical (Decision) Model Learning

In a nutshell:

EML = combinatorial problem + ML model

Main advantages:

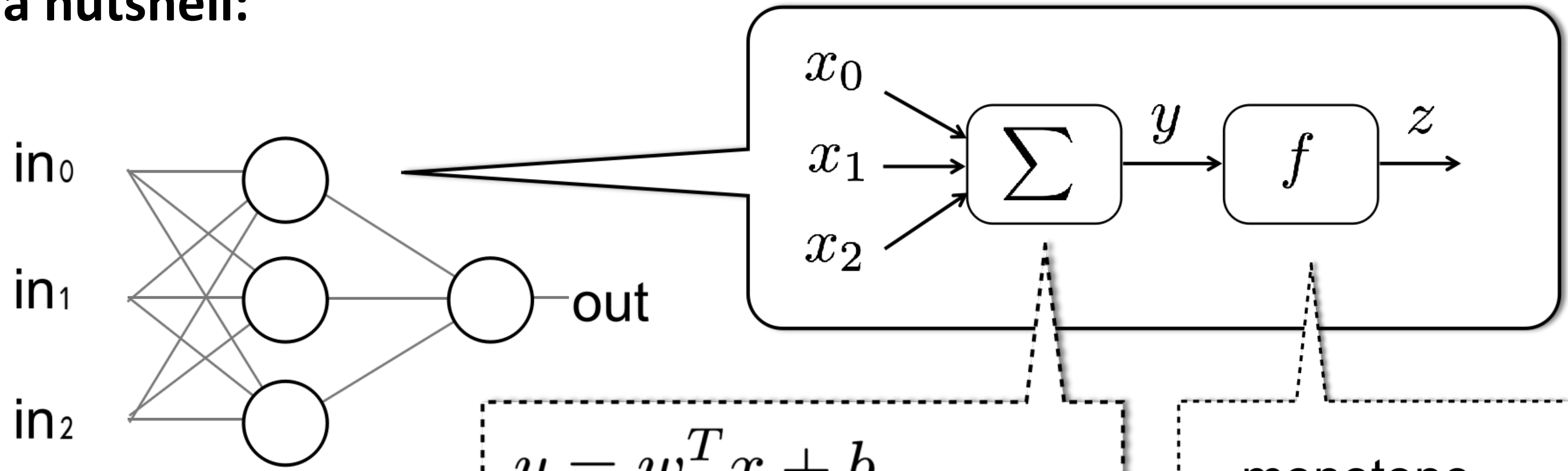
- Can deal with complex systems
- Support for complete search
- Declarative model
- Still benefits from bounding, propagation, conflict learning...

Embedding ML Models

The key step is “embedding” ML models
in declarative optimization

Neural Networks

In a nutshell:

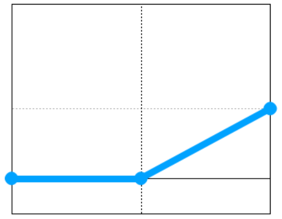


$$y = w^T x + b$$

$$z = f(y)$$

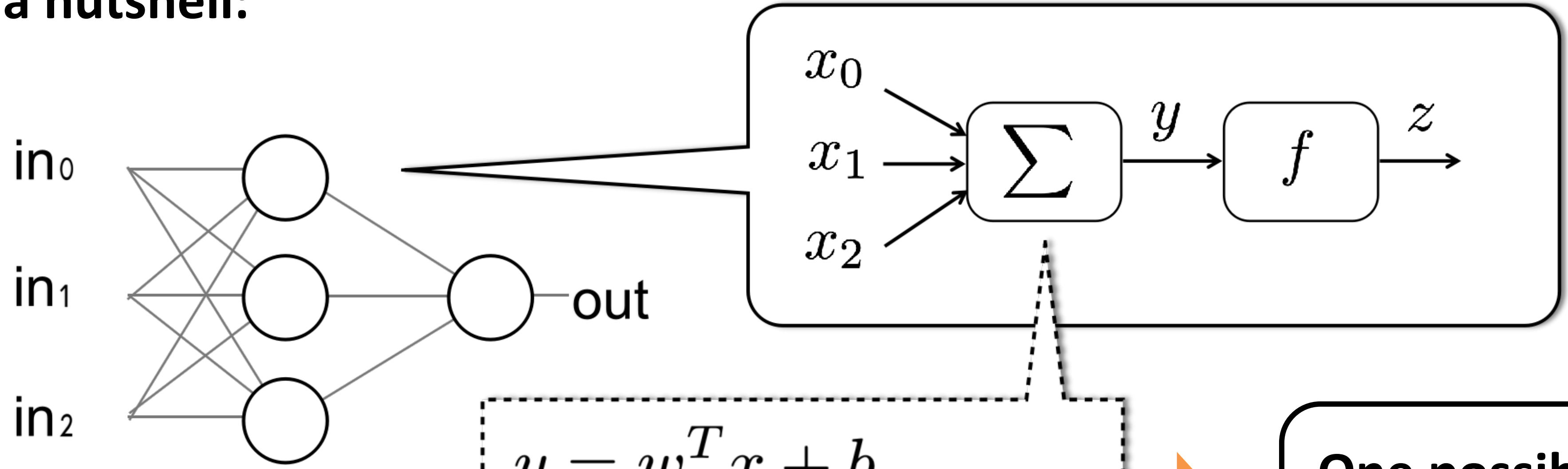
- monotone
- non-decreasing

e.g. ReLU



Neural Networks in CP

In a nutshell:



$$y = w^T x + b$$
$$z = f(y)$$

One possibility in CP: a global constraint per neuron

In a nutshell:

- ub(y) changes ↔ ub(z) changes
- lb(y) changes ↔ lb(z) changes

Neural Networks in CP

One possibility in CP: a global constraint per neuron

```
neuron_cst([X], Y, [W], b)
```

Input variables

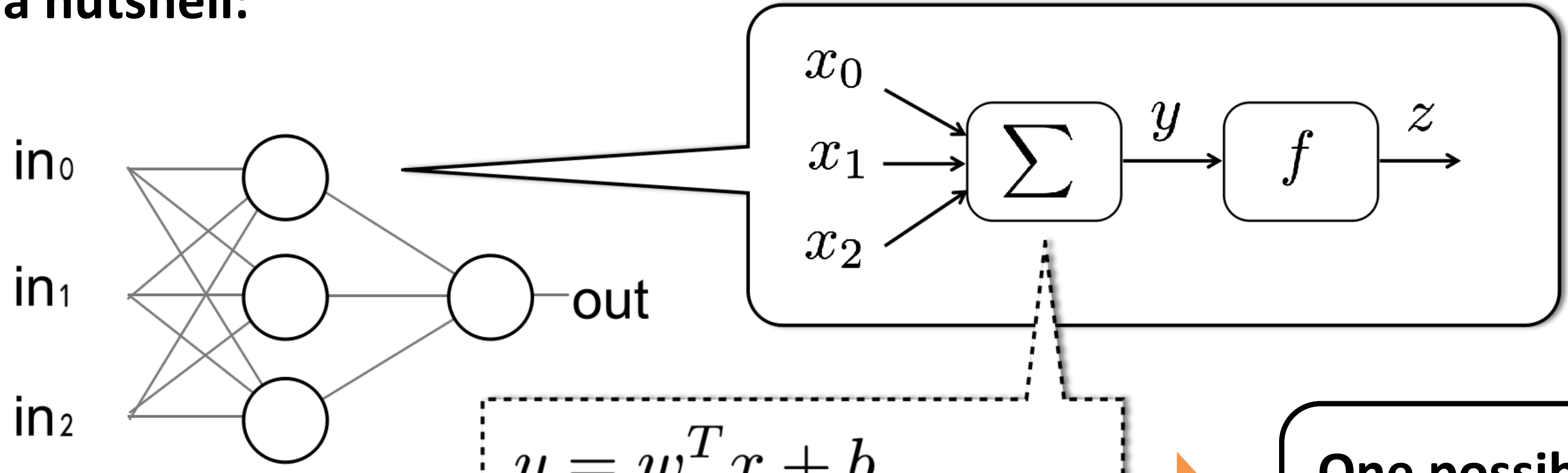
Output variable

Weights

Bias

Neural Networks in CP

In a nutshell:



$$y = w^T x + b$$
$$z = f(y)$$

One possibility in CP: a global constraint per neuron

Main drawback: local reasoning often results in weak bounds

In a nutshell:

- ub(y) changes ↔ ub(z) changes
- lb(y) changes ↔ lb(z) changes

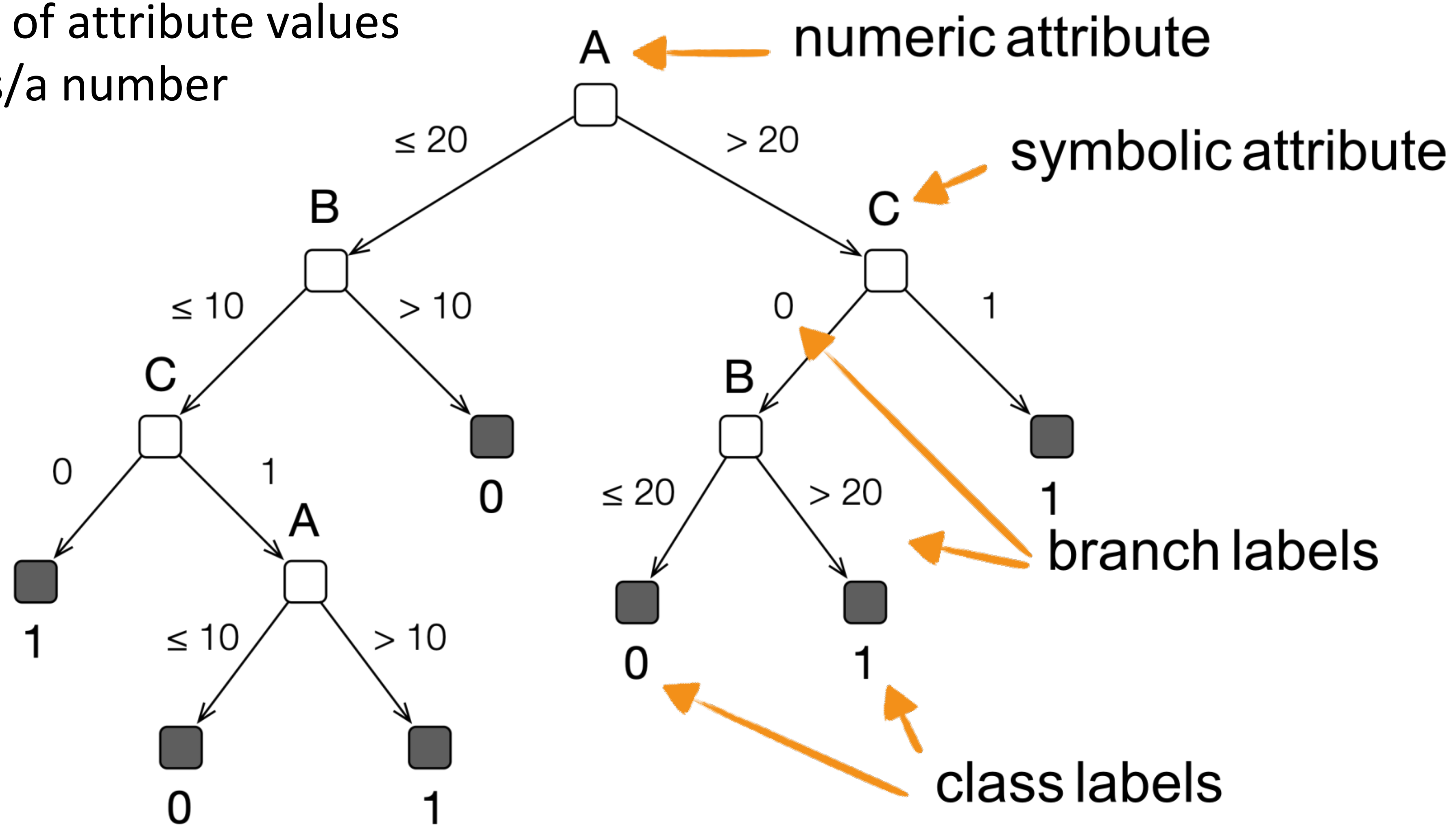
M. Lombardi, S. Gualandi: A lagrangian propagator for artificial neural networks in CP. Constraints 2016

Decision Trees

Bonfietti, Lombardi, Milano: Embedding Decision Trees and Random Forests in CP, CPAIOR 2011

Some experimental results

- Input: a vector of attribute values
- Output: a class/a number



Decision Trees in CP

How do we embed a DT in CP?

- A decision variable for each attribute

  $A \in \{-inf, inf\}$

  $B \in \{-inf, inf\}$

  $C \in \{0, 1\}$

- A decision variable for the class

  $Y \in \{0, 1\}$

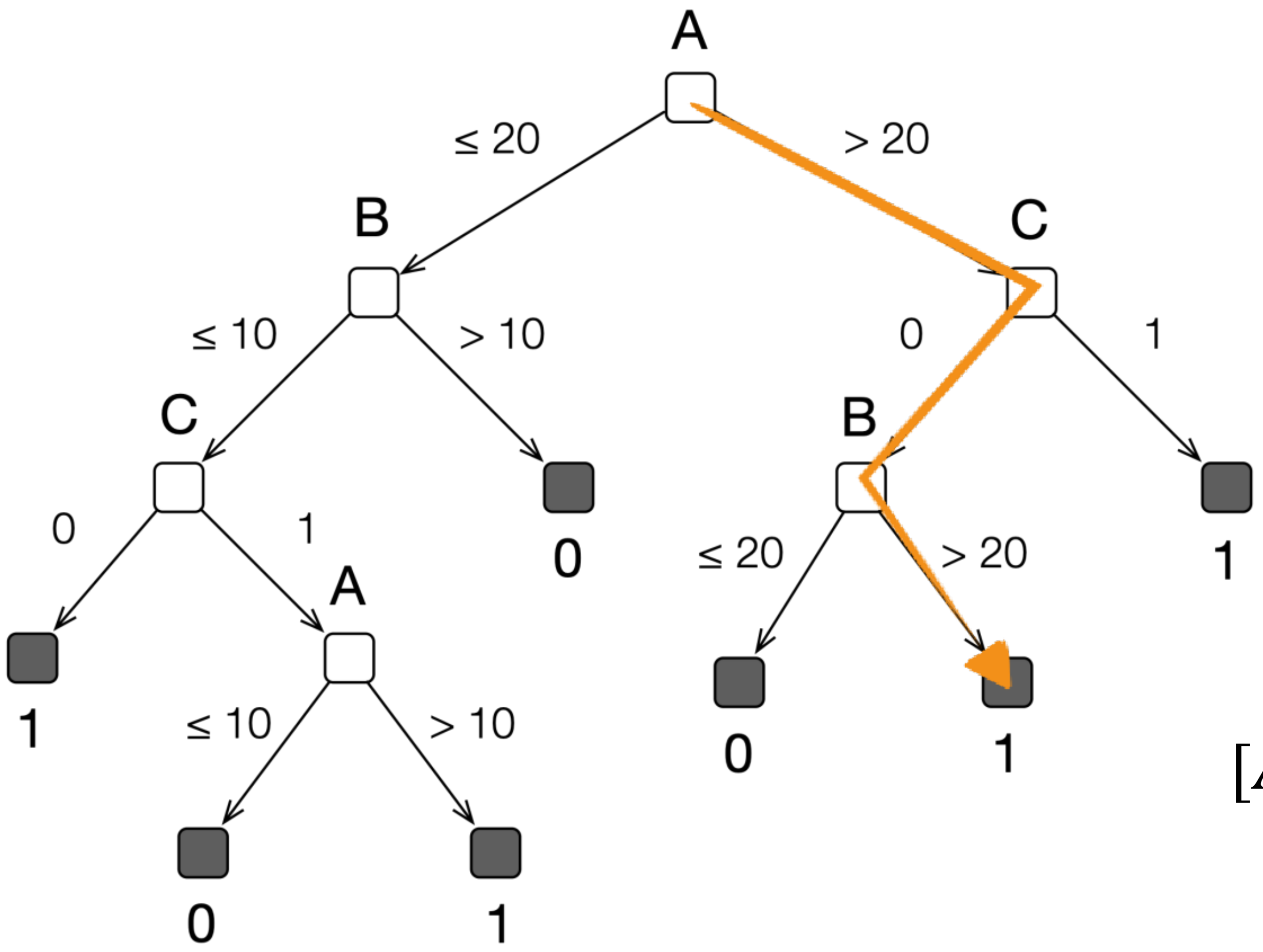
- Enforce consistency on:

  $Y = DT(A,B,C)$

This is the tricky part

Decision Trees in CP

A first, simple, encoding:



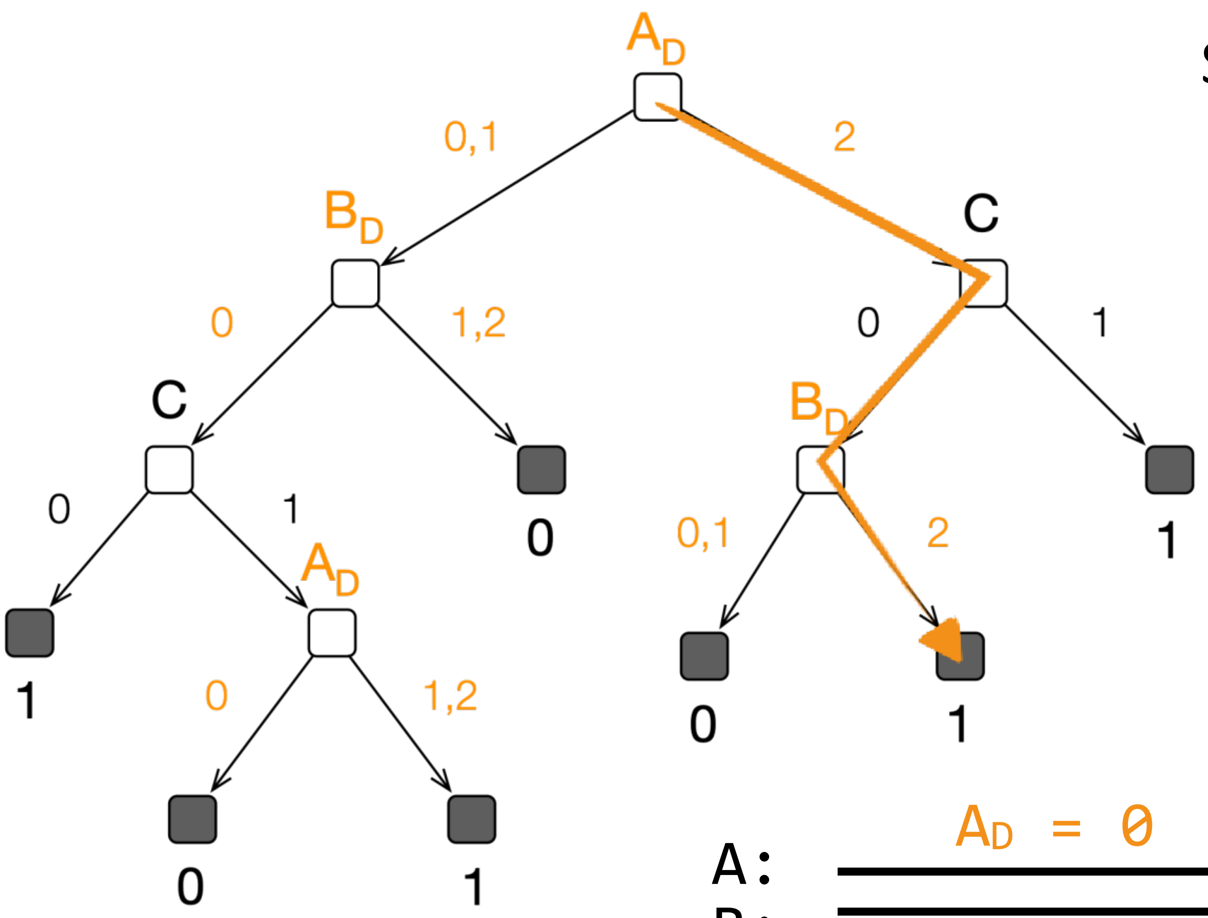
A path is an implication:

$$[A > 20] \wedge [C = 0] \wedge [B > 20] \;\Rightarrow\; [Y = 1]$$

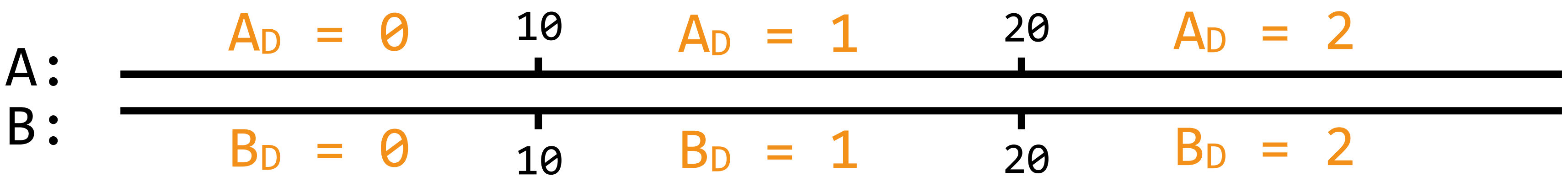
Decision Trees in CP

ALMA MATER STUDIORUM UNIVERSITÀ DI BOLOGNA

A second, stronger, encoding:

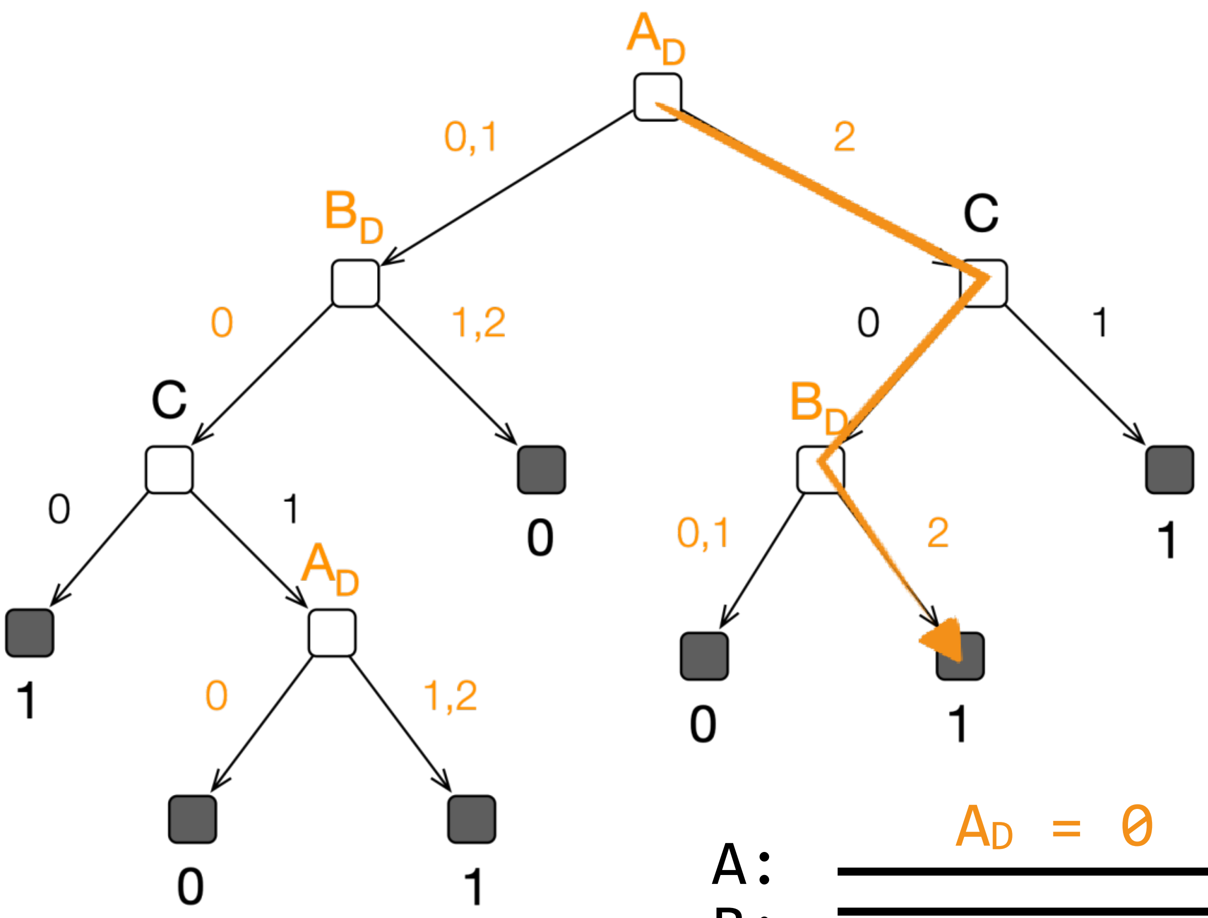


Step 2: all splits become over finite domain variables



Decision Trees in CP

A second, stronger, encoding:

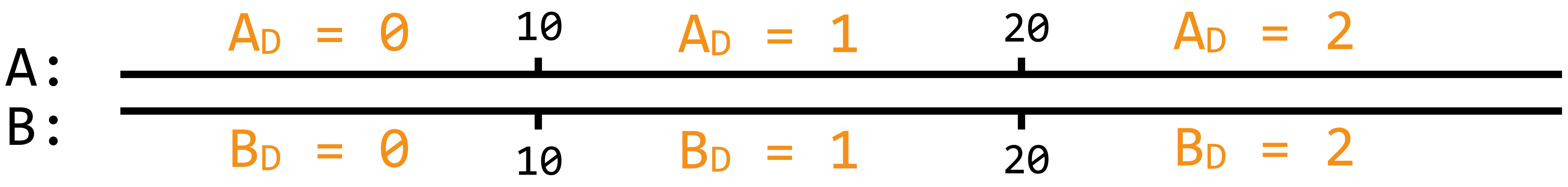


Step 3: a path is a set of feasible assignments

A_D	B_D	C	Y
2	2	0	1
…	…	…	…
…	…	…	…
…	…	…	…

Table constraint!

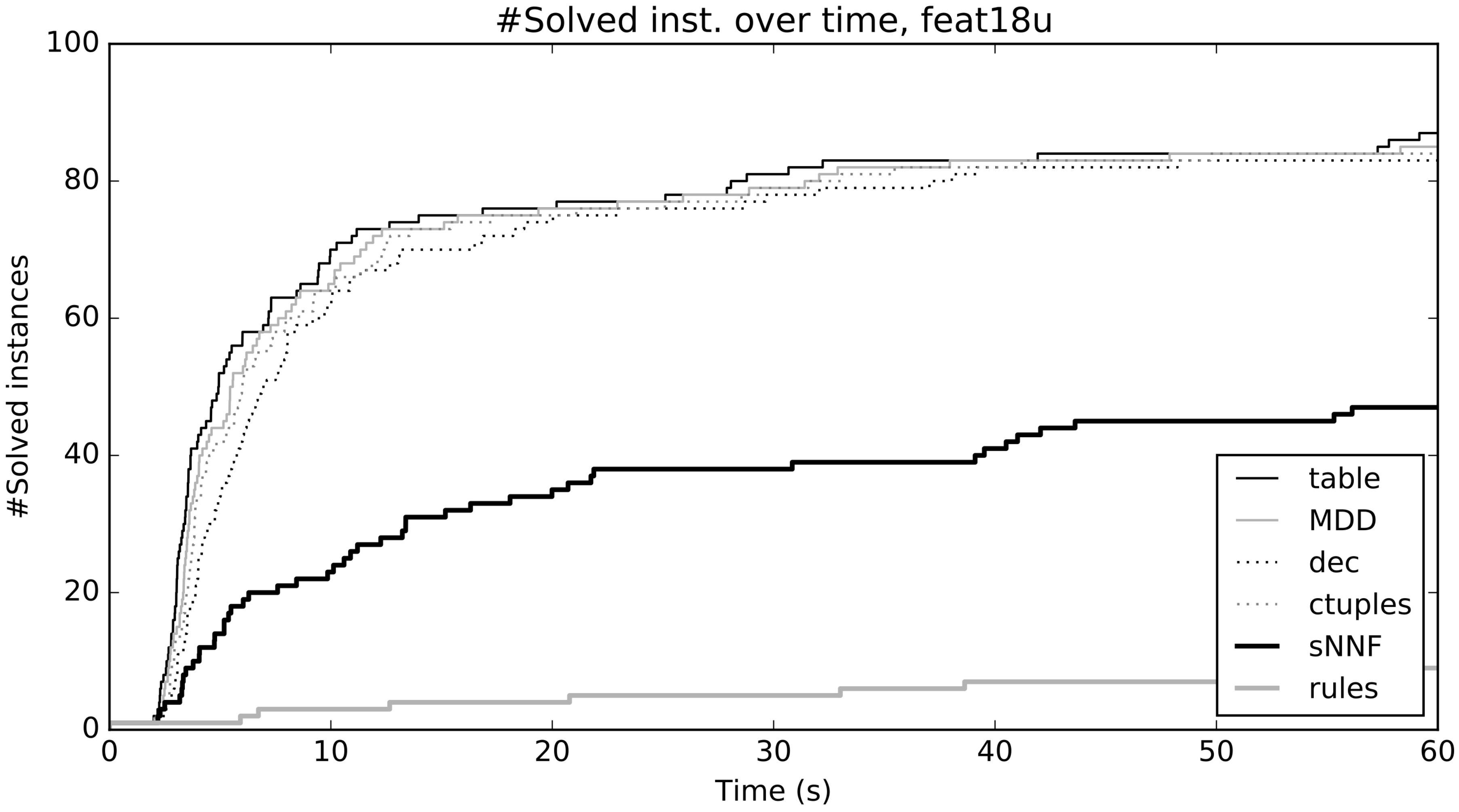
Bessiere, Regin, IJCAI 97



Decision Trees in CP

ALMA MATER STUDIORUM UNIVERSITÀ DI BOLOGNA

Some experimental results (including other encodings)

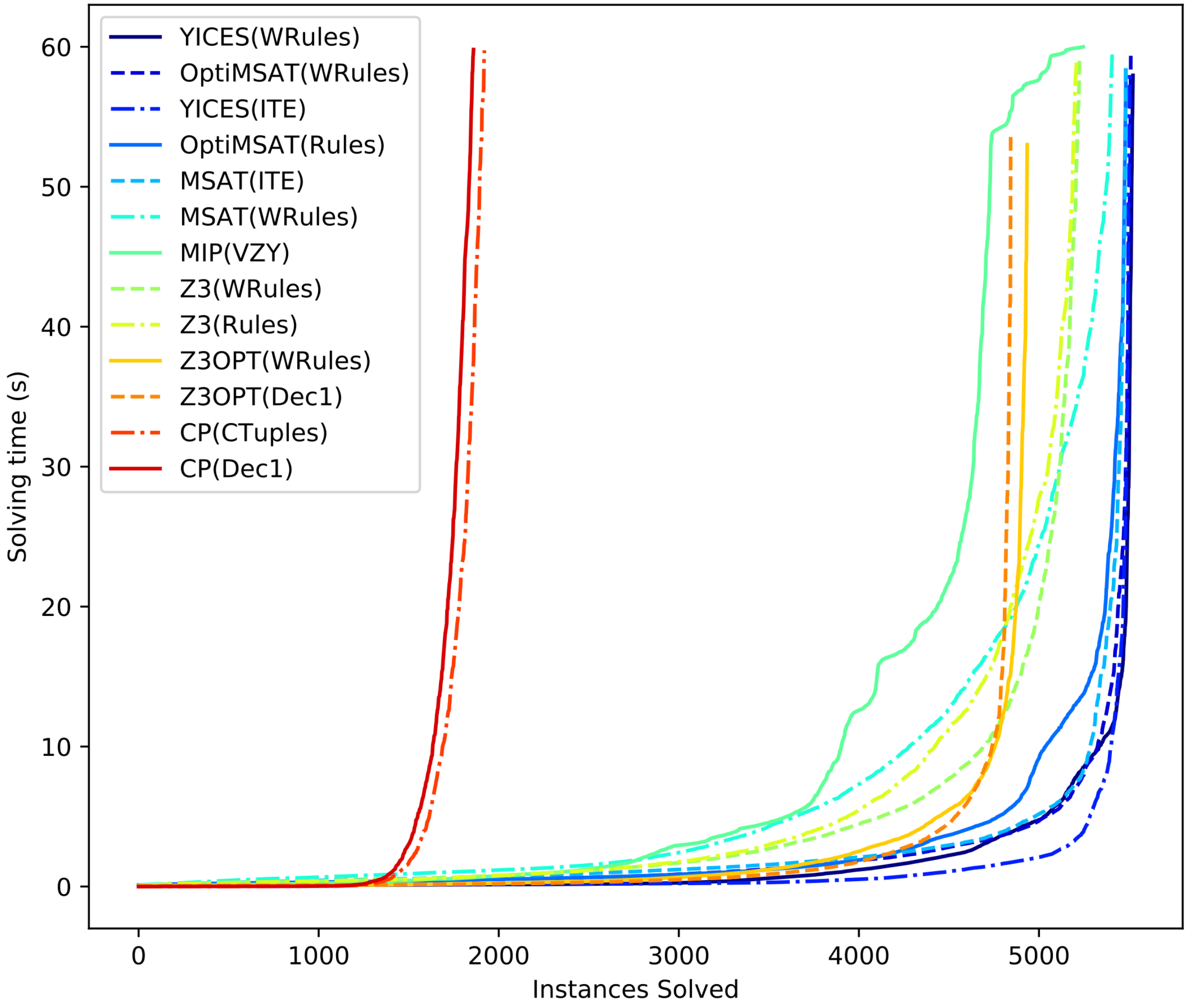


Decision Trees in SMT

In SAT Modulo Theories:

- Same encodings as in CP
- ...Except for those based on the TABLE cst
- But we have conflict learning

Some experimental results



Related Approaches

EML is strongly related to several other fields/techniques

- Black-box optimization (with surrogate models)
- System identification
- Local search/GAs + actual simulation

Some resources: http://emlopt.github.io

1. Papers
2. Running survey
3. EMLlib embedding techniques
4. Pre- and post- processing methods
5. I/O support (in particular readers for popular ML libraries)
6. Tutorial with an hands-on example

ALMA MATER STUDIORUM
UNIVERSITÀ DI BOLOGNA

EML Applications

EML Applications

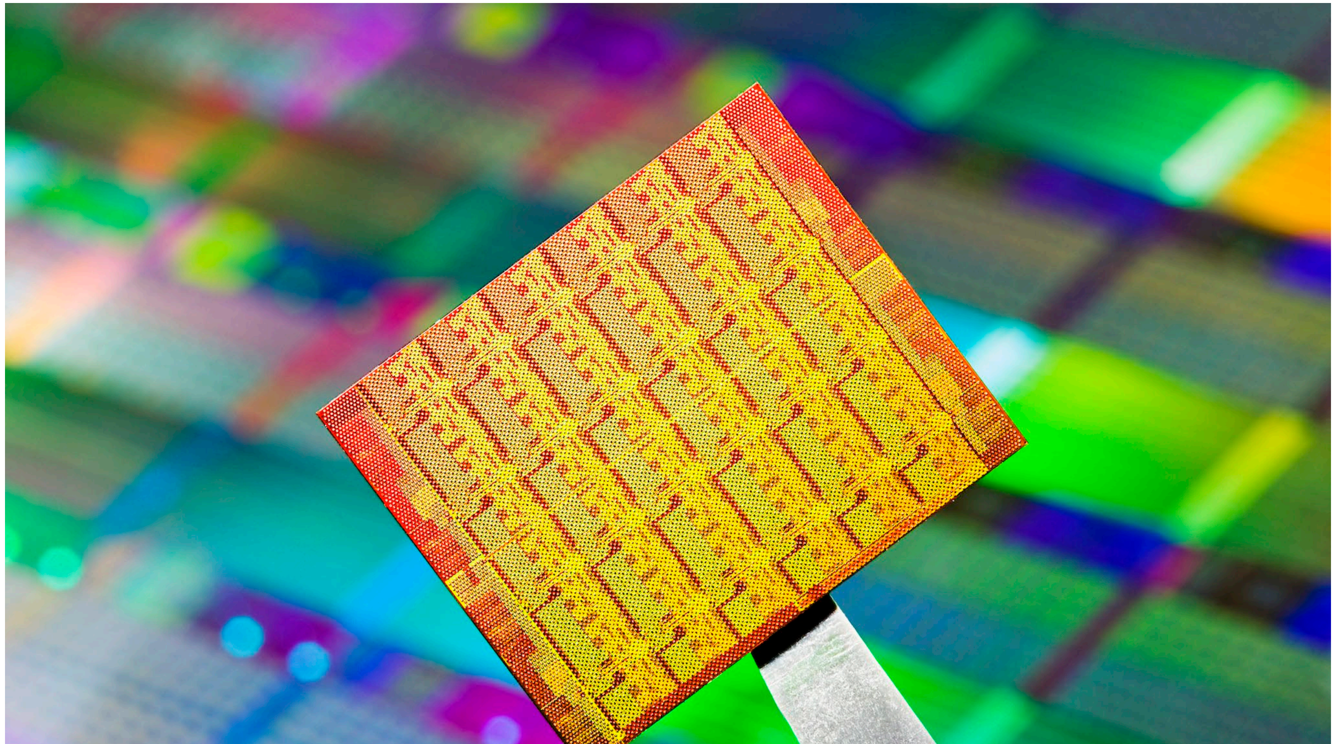
- EML has been used in different and diverse contexts

- Some success stories:
  - Thermal Control in System-on-Chips processors
  - Transprecision Computing
  - Epidemiological models
  - Hardware dimensioning and calibration of anticipatory algorithms
  - Verification & Adversarial examples

Katz, Guy, et al. "Reluplex: An efficient SMT solver for verifying deep neural networks." Int. Conf. on Computer Aided Verification, 2017.

Thermal Aware Job Allocation

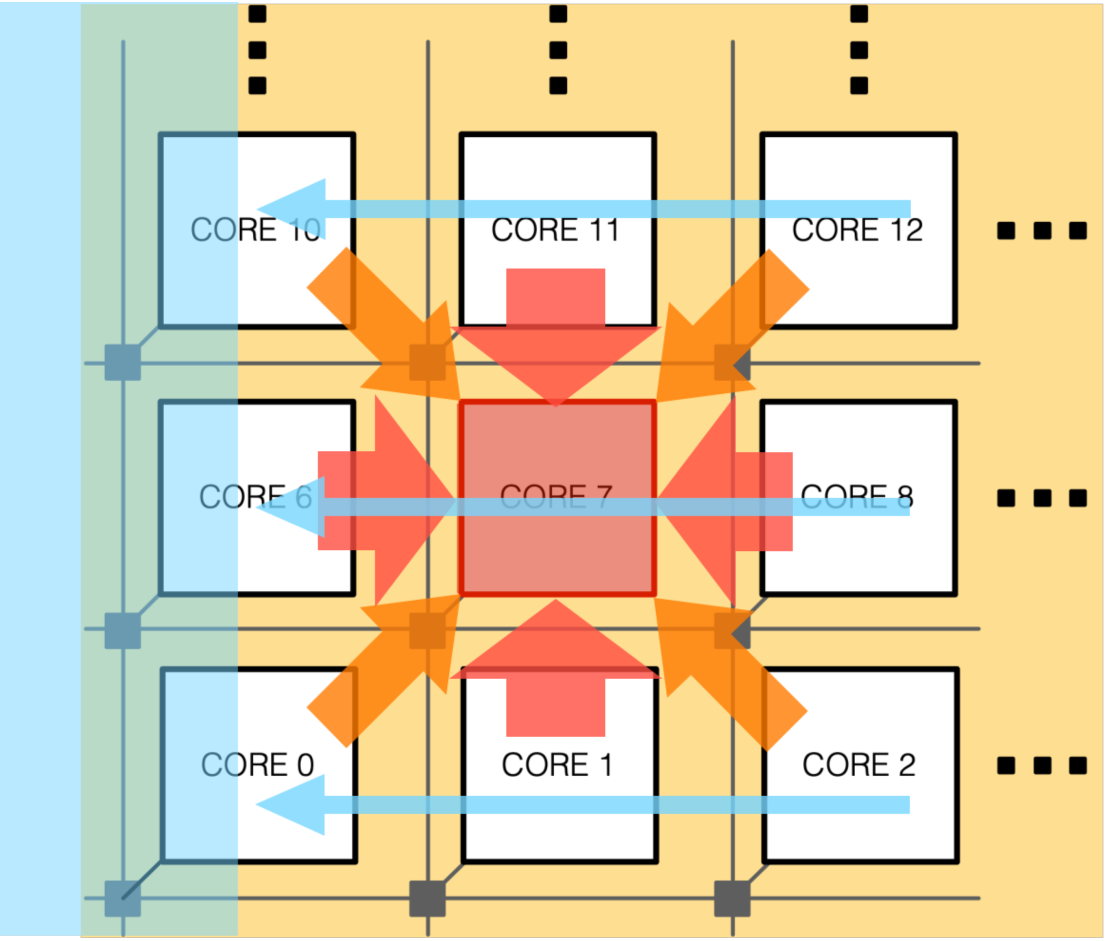
- Many-core CPU (Intel SCC, 2009, 48 cores, Xeon Phi precursor)
- Dispatch jobs
- Load balancing constraints
- **Objective:** avoid thermal hot-spots (efficiency loss)



Thermal Aware Job Allocation

ALMA MATER STUDIORUM UNIVERSITÀ DI BOLOGNA

The temperature/efficiency of a core is affected by:

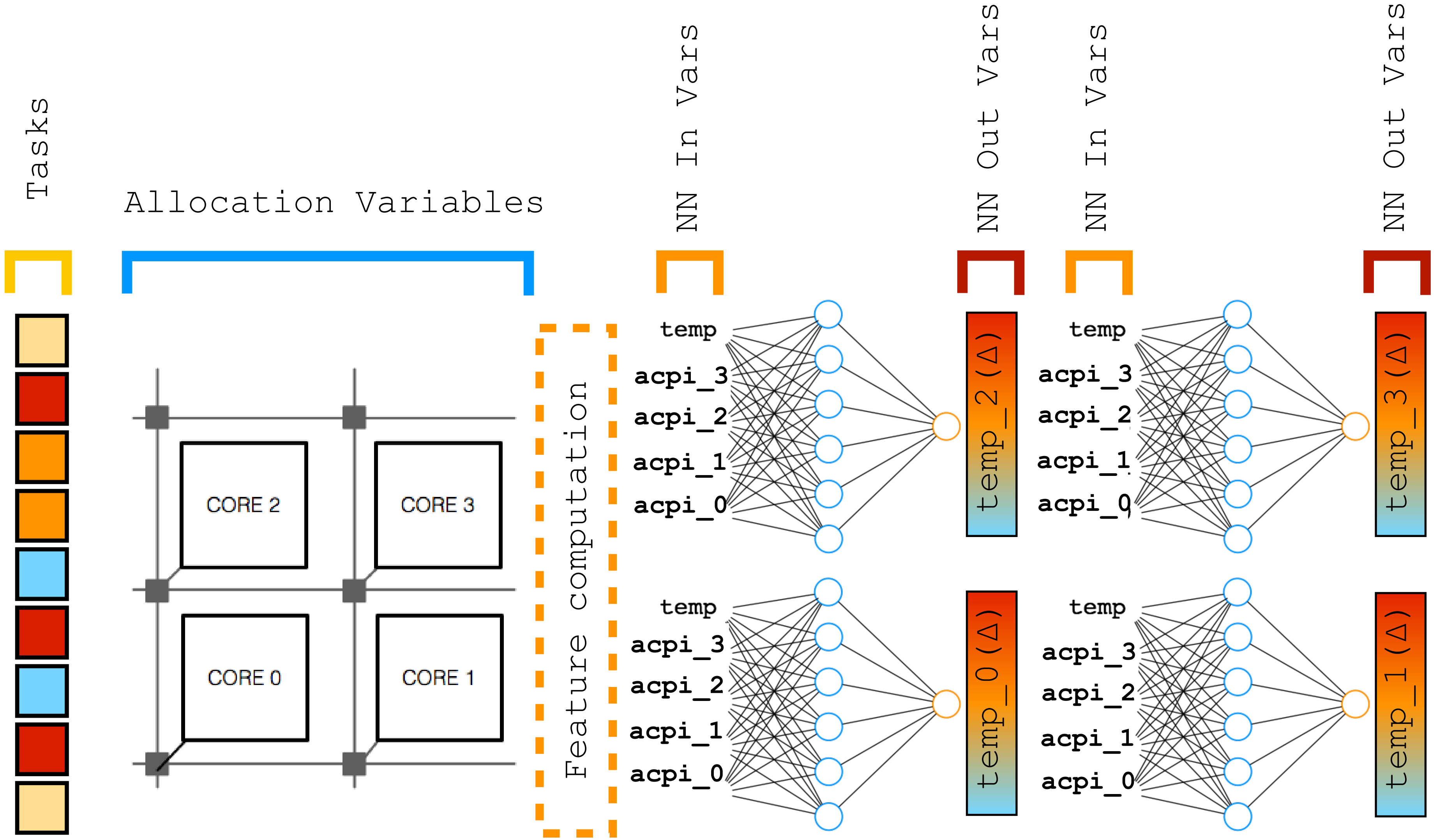


- the room temperature
- the workload of each core
- the neighbor workload
- the heat sink positions...

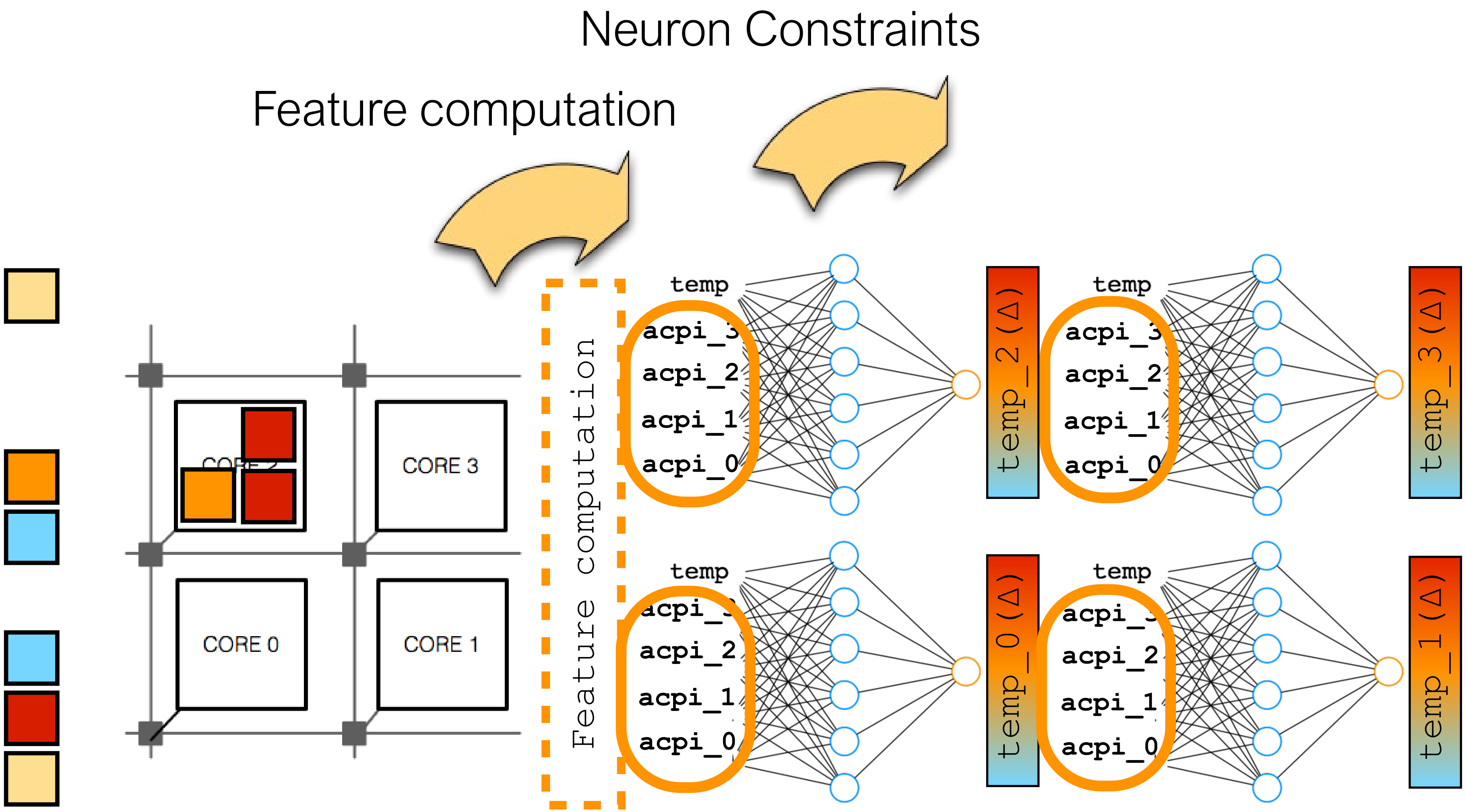
A simulator is viable, but
not so a declarative model

Sometimes, you don't even have a simulator

How it works at search time

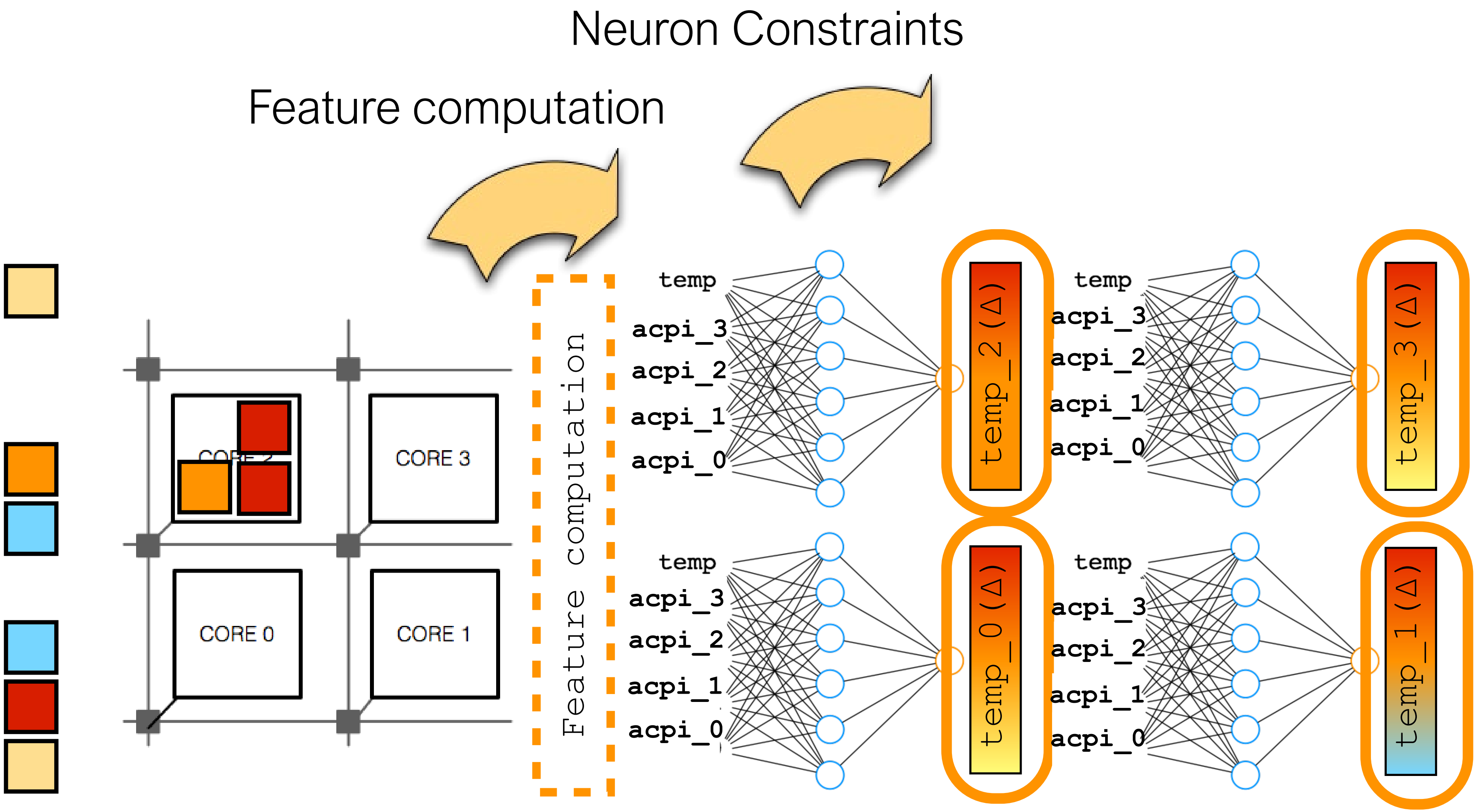


How it works at search time

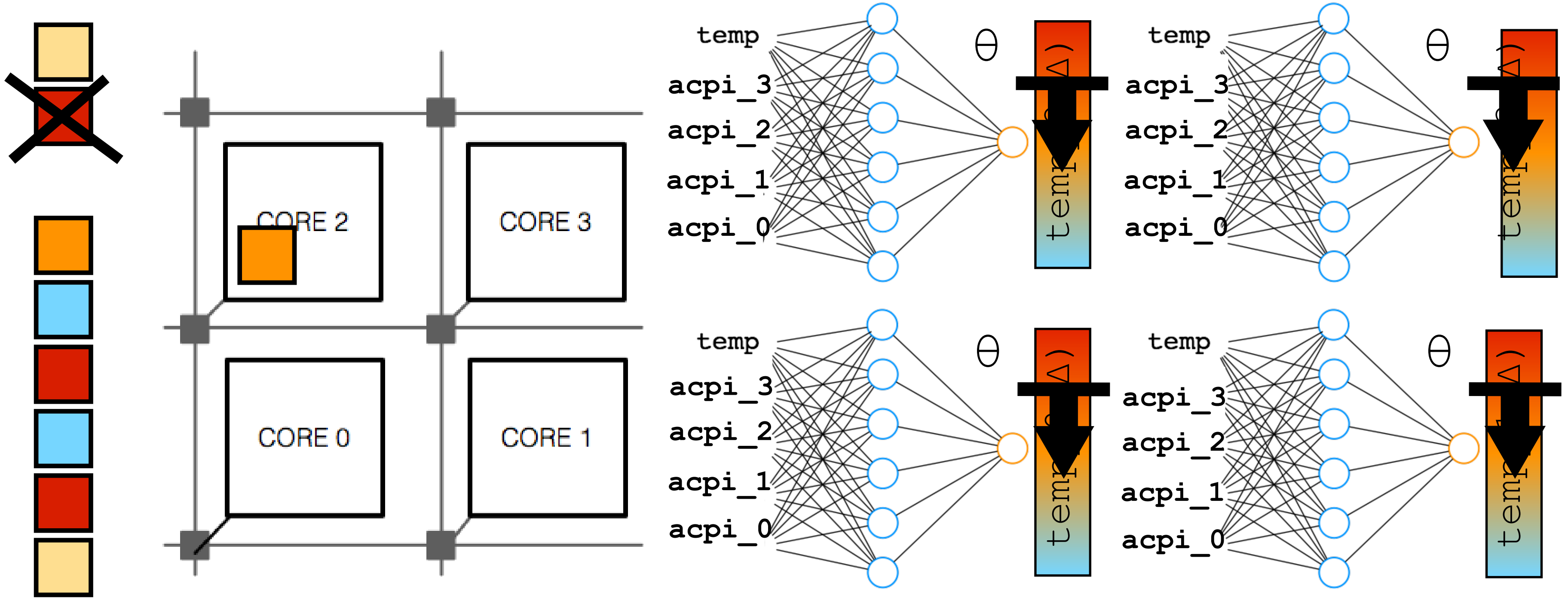


Forward propagation

ALMA MATER STUDIORUM
UNIVERSITÀ DI BOLOGNA

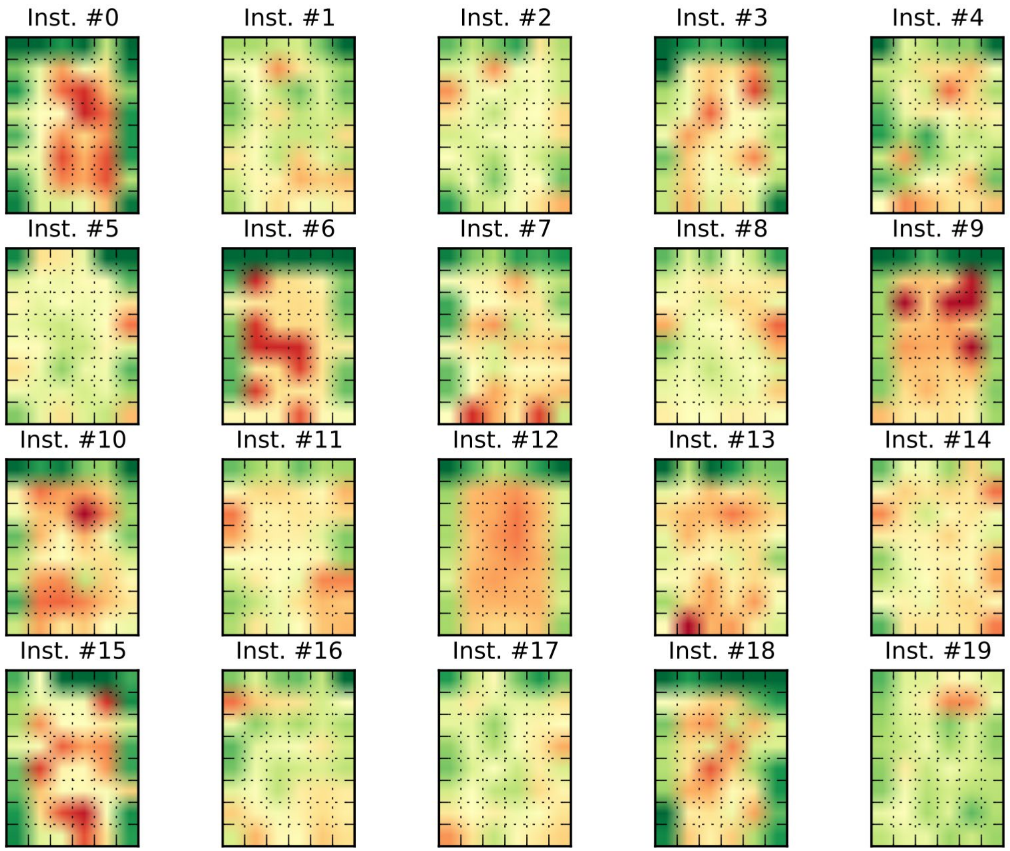


Backward propagation

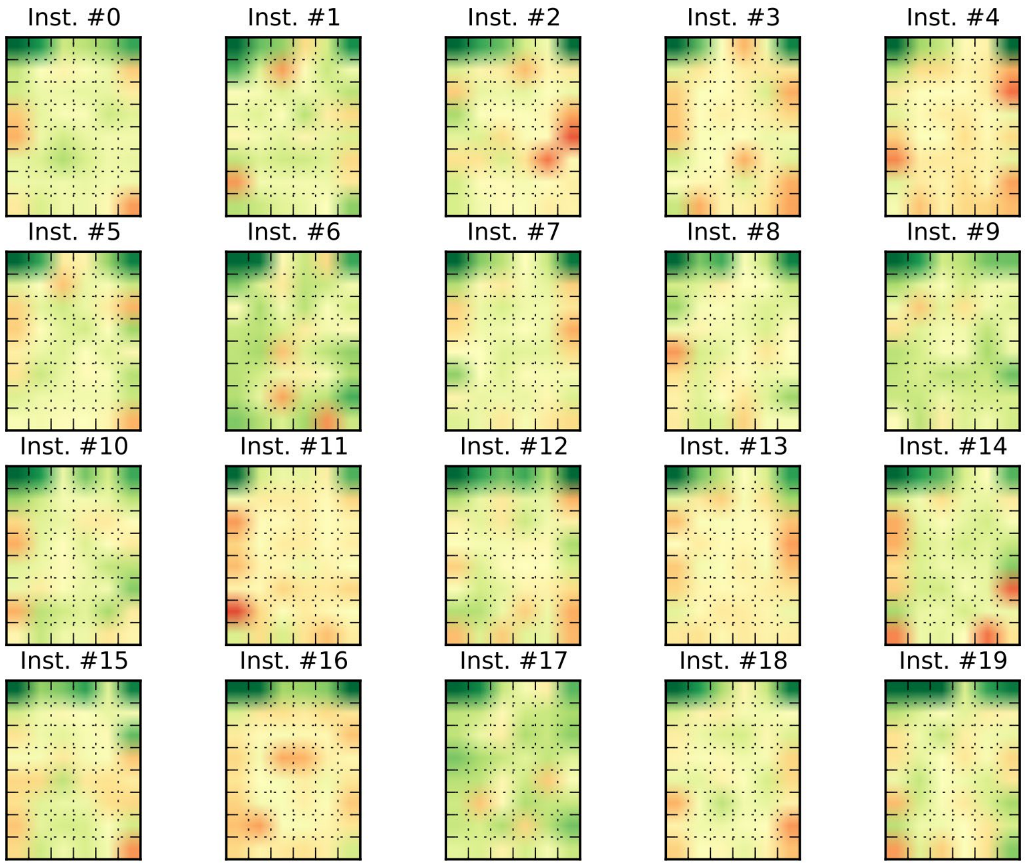


Thermal Aware Job Allocation

The simulated core efficiencies



Optimal solution with a linear model



60s LNS with CP (csts on single neurons)

EML - Transprecision Computing

- **Transprecision computing** is a paradigm to control approximation in space and time **at a fine grain**
- E.g. programs are written using standard FP formats
  - C/C++ programs → float and double variables
  - Precision tuning → transforming programs by changing default FP types to introduce smaller ones
- We have a benchmark with N variables
  - To each variable we can assign a number of bits
  - Different configurations lead to different errors
  - We want to minimize the number of bits assigned to each variable while respecting a constraint on the error
- **Optimization Problem**
  - We can model this problem as a MP (Mathematical Programming) problem

EML - Transprecision Computing

- Problem: ideally we would like to know the relationship between a bit configuration and the error
- BUT this relationship is *complex and not known, nor analytically expressible*

- Idea: Apply Machine Learning techniques to learn the relationship

Objective:

min (sum(B))

Constraints (simplified):

ML_R(B) <= E_target

ML_C(B) != (large_error)

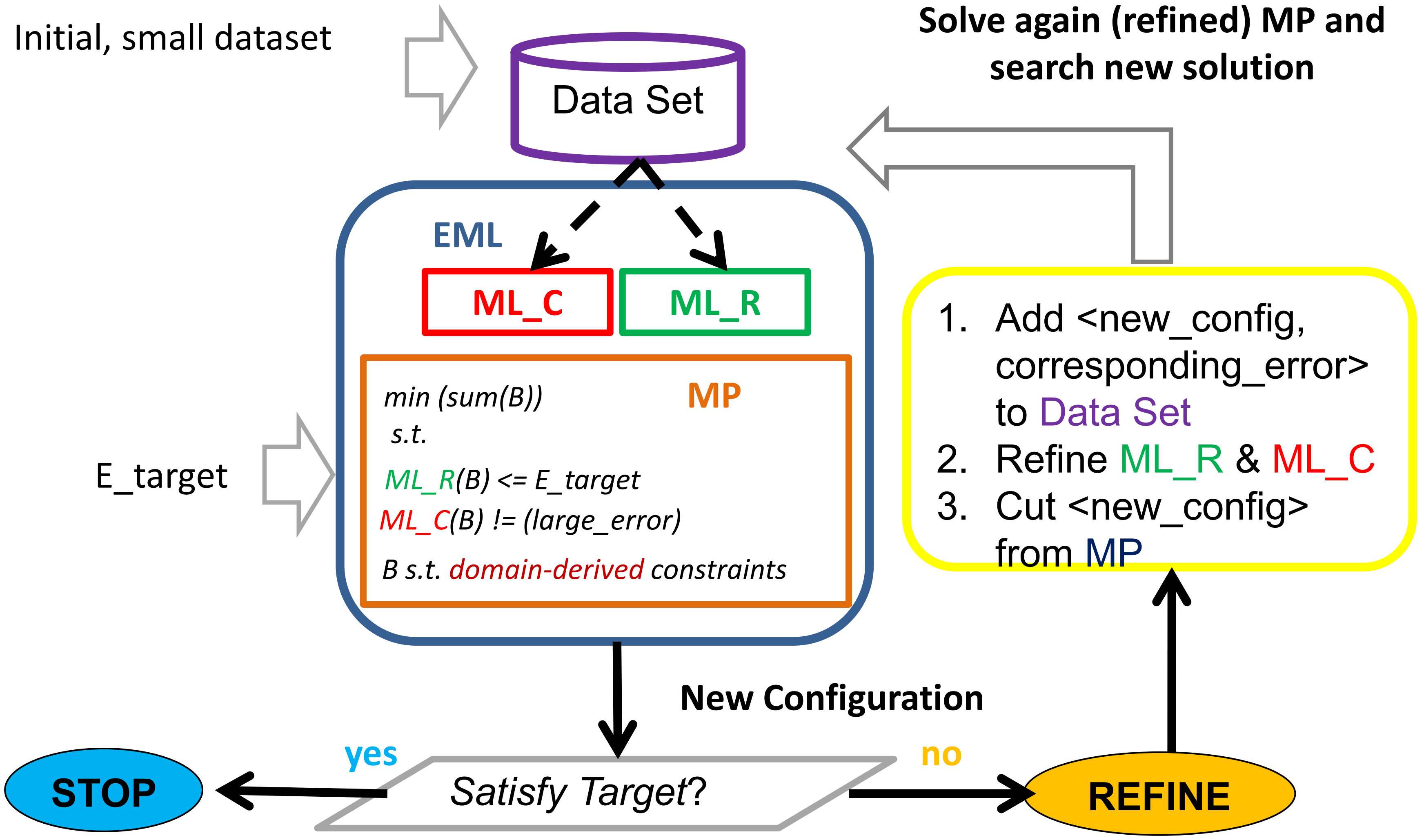
The predicted error for the bit config must be smaller than the target

The error associated to the bit config must be small (redundant constraint to increase robustness)

EML - Transprecision Computing

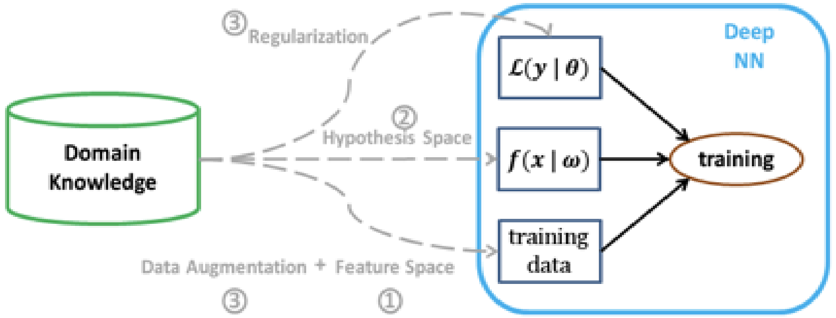
- Embed the learned relationships as a set of linear and non-linear constraints

- **What happens if the ML model prediction is not accurate?**
  - **Refine the model** and search for a new solution (similar to *Active Learning*)



Improve the model

Approximation of the function describing the relation precision-accuracy through **domain knowledge injection in deep neural network**



1. **Feature Addition –** create new features in the training set based on the domain knowledge available
2. **Ad-hoc Network Topology –** the relation between the variable ca encoded through graphs/networks
3. **Regularization function and Data augmentation –** enforce properties (*constraints*) during the training

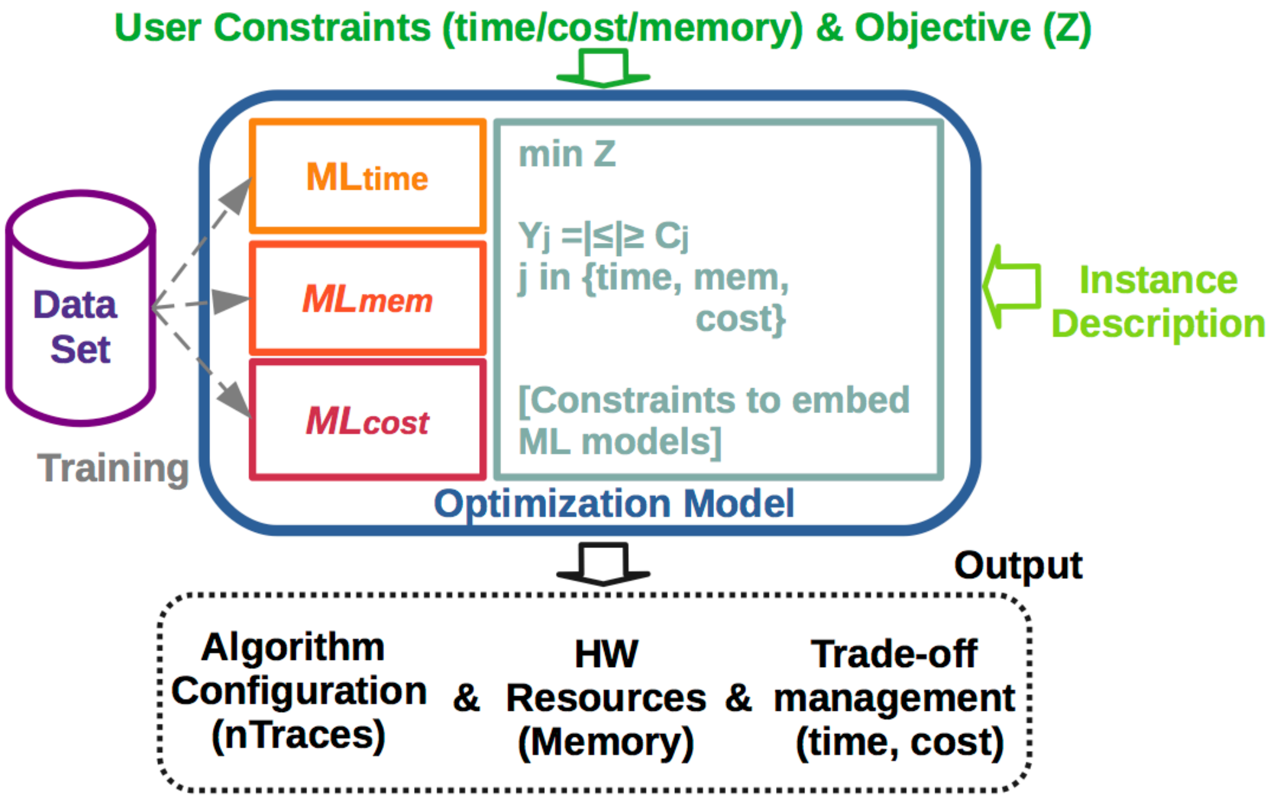
EML - HW dimensioning

- Problem: Given an AI algorithm or an AI tool, which is the best hardware configuration to satisfy time/QoS/cost constraints

- Idea: apply Machine Learning techniques to learn the relationship between hardware configuration parameters and algorithm performances (i.e., runtime, memory usage, solution quality)
  - Then embed ML models inside an optimization model that allows to impose also user requirements hardware constraints

EML - HW dimensioning & configuration of anticipatory algs.

- The optimization model takes as input:
  - **user-defined constraints** and an **objective goal** (e.g., minimizing the alg. runtime)
  - a **new and unseen instance description**
  - **dataset to train** three **ML regression models**, each one predicts a specific target:
    - the time required by the online algorithm to find a solution (MLtime);
    - the amount of memory (MLmem);
    - the solution quality, expressed in terms of its cost (MLcost)



- It produces as output:
  - the **optimal matching** among algorithm configuration, hardware resources and time/solution trade-off

EML Epidemiological Models

- Development of a **Decision Support System** to predict the spreading of a virus and prescribe Intervention Plans to minimize both infected cases and socio-economic side effects
  - XPrize Pandemic Response Challenge for COVID19

- Predictive Model:
  - LSTM
  - Compartimental Models (e.g. SIR) + ANN

- Combinatorial Model specifying and objective and constraints w.r.t. the problem

F.Baldo, M.Iannello, M.Lombardi, M.MIlano:
Informed Deep Learning for Epidemics Forecasting. PAIS@ECAI 2022:

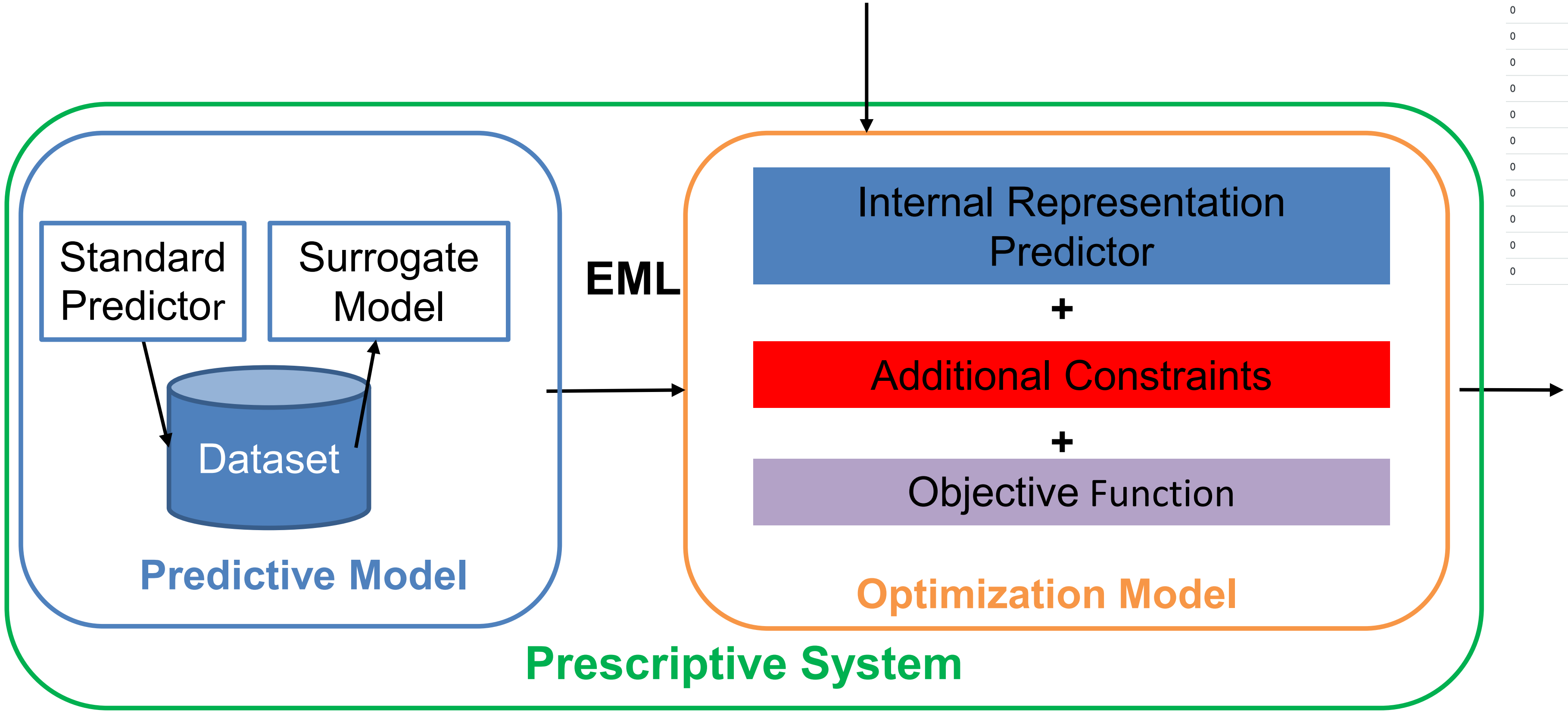
EML Epidemiological Models

ALMA MATER STUDIORUM
UNIVERSITÀ DI BOLOGNA

CountryName	RegionName	C1_School closing	C2_Workplace closing	C3_Cancel public events
Afghanistan		0.83	1.71	1.44
Albania		0.14	1.44	0.1
Algeria		0.06	0.13	0.55
Andorra		0.33	1.56	1.45
Angola		1.01	0.51	0.76

```
Italy,ITA,,,NAT_TOTAL,20201226,3.0,1.0,3.0,1.0,2.0,1.0,4.0,1.0,1.0,
Italy,ITA,,,NAT_TOTAL,20201227,3.0,1.0,3.0,1.0,2.0,1.0,4.0,1.0,1.0,
Italy,ITA,,,NAT_TOTAL,20201228,3.0,1.0,2.0,1.0,2.0,1.0,4.0,1.0,1.0,
Italy,ITA,,,NAT_TOTAL,20201229,3.0,1.0,2.0,1.0,2.0,1.0,4.0,1.0,1.0,
Italy,ITA,,,NAT_TOTAL,20201230,3.0,1.0,2.0,1.0,2.0,1.0,4.0,1.0,1.0,
Italy,ITA,,,NAT_TOTAL,20201231,3.0,1.0,3.0,1.0,2.0,1.0,4.0,1.0,1.0,
Italy,ITA,,,NAT_TOTAL,20210101,3.0,1.0,3.0,1.0,2.0,1.0,4.0,1.0,1.0,
Italy,ITA,,,NAT_TOTAL,20210102,3.0,1.0,3.0,1.0,2.0,1.0,4.0,1.0,1.0,
Italy,ITA,,,NAT_TOTAL,20210103,3.0,1.0,3.0,1.0,2.0,1.0,4.0,1.0,1.0,
Italy,ITA,,,NAT_TOTAL,20210104,3.0,1.0,3.0,1.0,2.0,1.0,4.0,1.0,1.0,
Italy,ITA,,,NAT_TOTAL,20210105,3.0,1.0,3.0,1.0,2.0,1.0,4.0,1.0,1.0,
Italy,ITA,,,NAT_TOTAL,20210106,3.0,1.0,3.0,1.0,2.0,1.0,4.0,1.0,1.0,
Italy,ITA,,,NAT_TOTAL,20210107,3.0,1.0,3.0,1.0,2.0,1.0,4.0,1.0,1.0,
Italy,ITA,,,NAT_TOTAL,20210108,3.0,1.0,3.0,1.0,2.0,1.0,4.0,1.0,1.0,
Italy,ITA,,,NAT_TOTAL,20210109,3.0,1.0,3.0,1.0,2.0,1.0,4.0,1.0,1.0,
Italy,ITA,,,NAT_TOTAL,20210110,3.0,1.0,3.0,1.0,2.0,1.0,4.0,1.0,1.0,
```

Interventions Cost + Historical Intervention Plan



PrescriptionIndex	CountryName	RegionName	Date	C1_School closing	C2_Workplace closing	C3_Cancel public events
0	Aruba		2020-08-01	0	1	1
0	Aruba		2020-08-02	0	2	0
0	Aruba		2020-08-03	0	0	1
0	Aruba		2020-08-04	1	2	1
0	Afghanistan		2020-08-01	1	1	1
0	Afghanistan		2020-08-02	2	1	0
0	Afghanistan		2020-08-03	2	1	1
0	Afghanistan		2020-08-04	1	2	1
0	Angola		2020-08-01	1	0	1
0	Angola		2020-08-02	0	0	0
0	Angola		2020-08-03	1	1	1
0	Angola		2020-08-04	1	2	0
0	Albania		2020-08-01	2	0	1

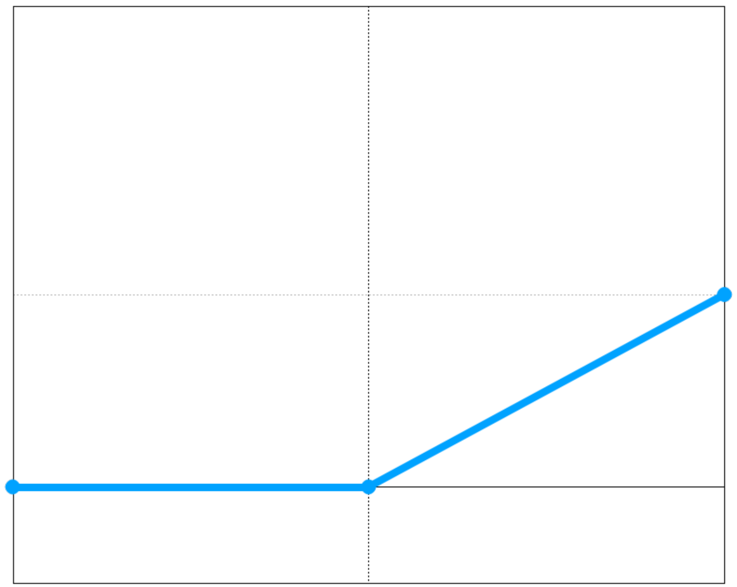
Intervention Plan

ALMA MATER STUDIORUM
UNIVERSITÀ DI BOLOGNA

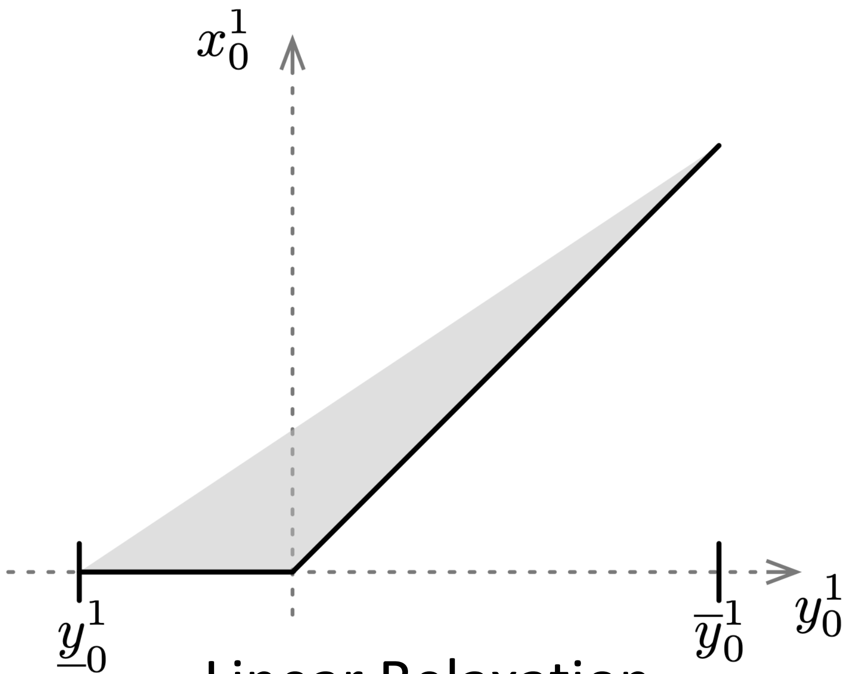
Open Issues, Open Directions

Weakening Relaxation and Large Models

The devil is in ~~the details~~ the bounds



True ReLU



Linear Relaxation

There is a trade-off:

- Poor bounds = poor relaxation
- Good bounds = expensive pre-processing

Weakening Relaxation and Large Models

ALMA MATER STUDIORUM UNIVERSITÀ DI BOLOGNA

The bottom line is that dealing with large ML models is hard

In the case of Neural Networks:

- Relaxations become exponentially weaker with depth
- Individual fully connected layers have dense coefficient matrix (hard for MILP)
- Some interesting progress in https://arxiv.org/pdf/2101.12708.pdf, but still unsolved
- Currently: strong bound tightening + MILP is still the best approach

In the case of Decision Trees:

- Individual trees may grow very large (many variables, many constraints)
- Relaxations for ensemble get weaker with the number of estimators
- Some progress on the OR side in *V. V. Misic. Optimization of Tree Ensembles. ´ Operations Research, 2020*

Accuracy vs Optimizability

In EML, higher accuracy is not always better!

Complex ML Models

- More accurate
- Run-time overhead
- Weaker inference (bounds, etc.)

Risk: poor optimization

Simple ML Models

- Less accurate
- Quicker to evaluate
- More effective inference

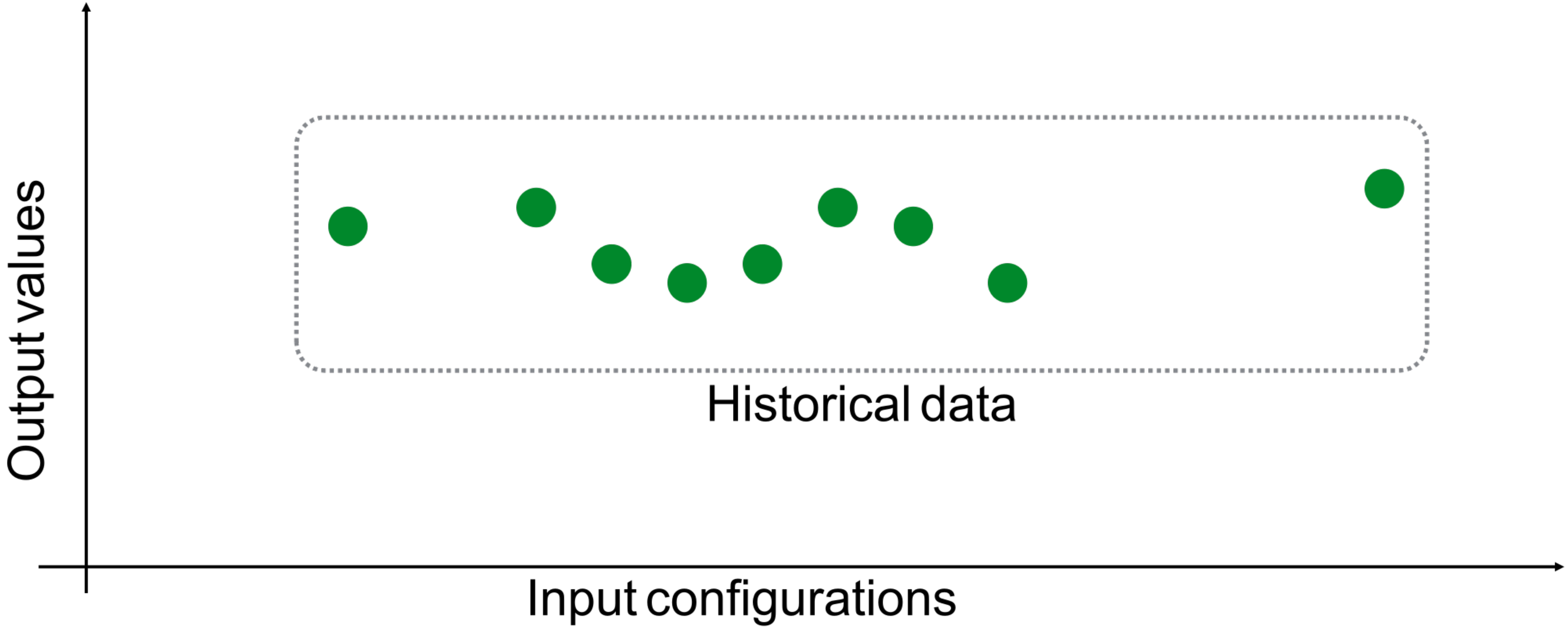
Risk: deceptively good solution

There is a trade off between model accuracy (variance) and optimization effectiveness

- How to characterize it?
- How to pick a model architecture?

Weak Spots in the ML Model

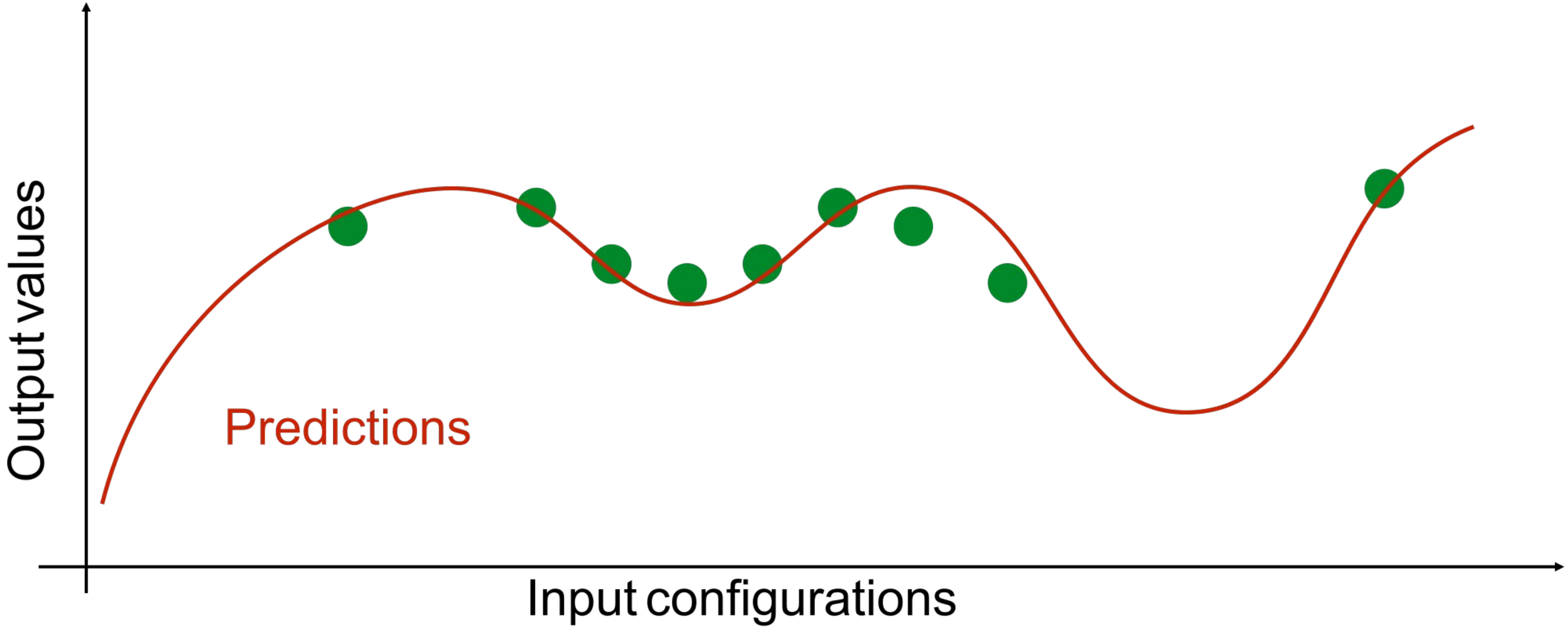
Say we have a training set that looks like this:



Weak Spots in the ML Model

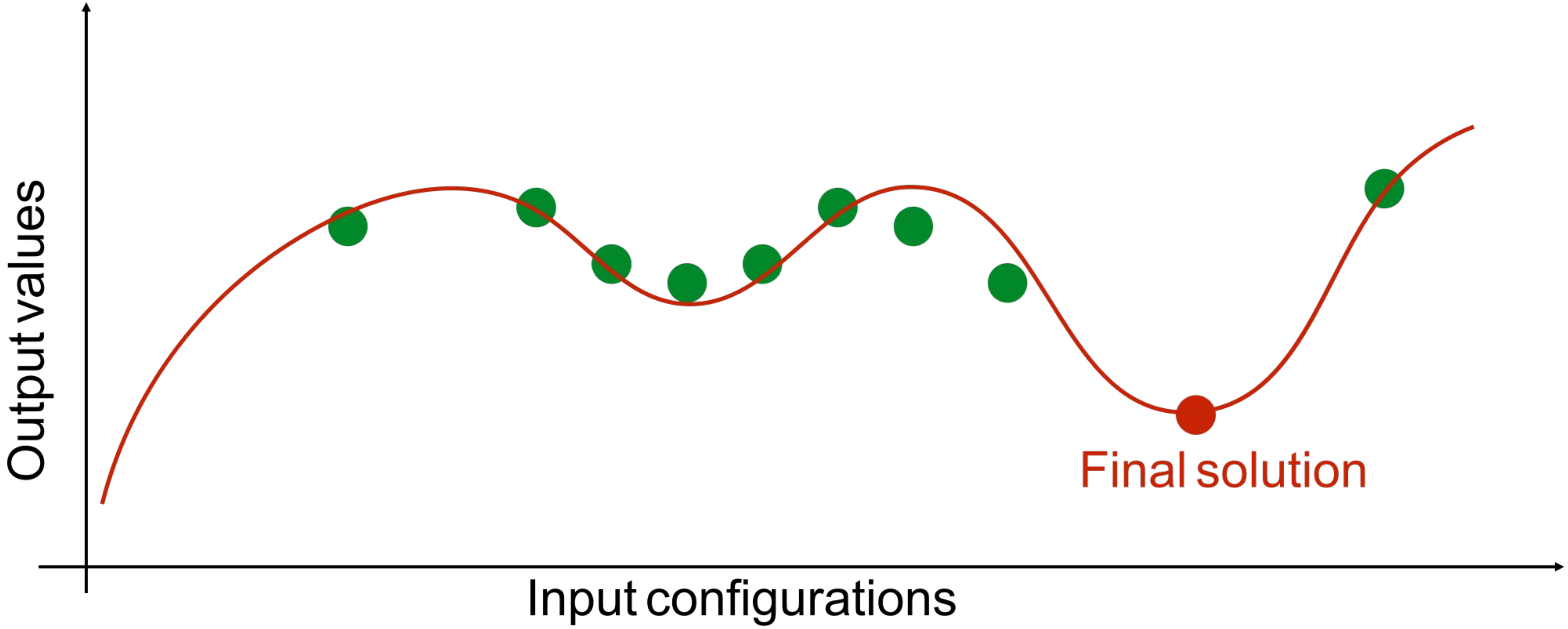
ALMA MATER STUDIORUM UNIVERSITÀ DI BOLOGNA

The ML model provides a prediction for each input value



Weak Spots in the ML Model

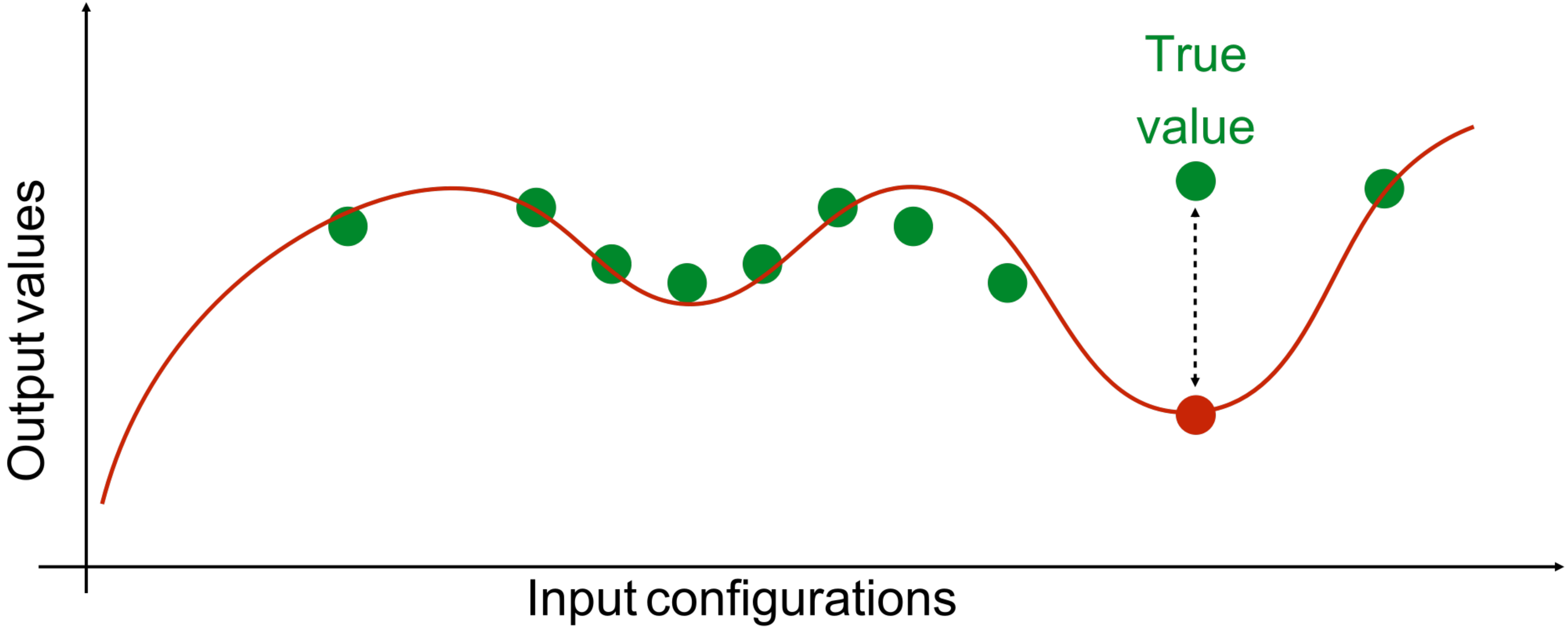
If (e.g.) the predicted value is the cost, the solver will seek to minimize it



Weak Spots in the ML Model

ALMA MATER STUDIORUM
UNIVERSITÀ DI BOLOGNA

If the solution if far from known points, there may be a large error



Weak Spots in the ML Model

What can be done?

- When building the training set
  - Factorial design, Latin hypercube sampling…
- At search time:
  - Active learning, if you can run experiments
  - Connection with preference elicitation and black box optimization

BB Optimization and Active Learning

ALMA MATER STUDIORUM
UNIVERSITÀ DI BOLOGNA

EML for Black-box optimization

- Conventional BB optimization approach rely on kernel-based approximation
- Their complexity grows with each sample
- In principle EML can do the same with a fixed size model, but:
- ...How to measure uncertainty on the unexplored areas?
- ...How to ensure meaningful ML model changes with each new sample?
- Some progress in https://arxiv.org/abs/2003.04774, but still very open

Formally Dealing with Uncertainty

ALMA MATER STUDIORUM UNIVERSITÀ DI BOLOGNA

ML Model are approximate

...But the level of approximation is quantifiable

- More than can be said for many expert-designed models
- Even more: probabilistic elements can be made part of the ML model
    - E.g. structured output representing the parameters of a known distribution
    - E.g. density estimators
- We could take advantage of this when doing optimization
- Possible applications: chance constraints without the usual additional complexity

Formally Dealing with Uncertainty

ML Model are approximate

...But the level of approximation is quantifiable

- E.g. probabilities in NN classifiers or Decision trees
- More than can be said for many expert-designed models
- Even more: probabilistic elements can be made part of the ML model
  - E.g. structured output representing the parameters of a known distribution
  - E.g. density estimators
- We could take advantage of this when doing optimization
- Possible applications: chance constraints without the usual additional complexity

Composition of Optimizers

ALMA MATER STUDIORUM
UNIVERSITÀ DI BOLOGNA

EML can enable optimization over complex systems

This includes controlled systems

- The ML model can learn the behavior of both the system and the optimizer
- An early, simple, example in: *Bartolini, et. al.: Optimization and Controlled Systems: A Case Study on Thermal Aware Workload Dispatching. AAAI 2012*

EML can be used to build hierarchies of optimizers

- We get integration without a feedback loop
- Trick: information exchange occurs at training time
- In principle: dramatically more scalable

ALMA MATER STUDIORUM
UNIVERSITÀ DI BOLOGNA

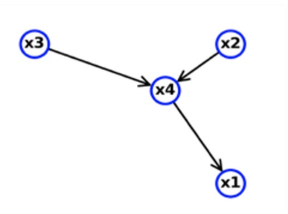
Thanks!

1. Feature Addition

- Additional features to *characterize the precision configurations*

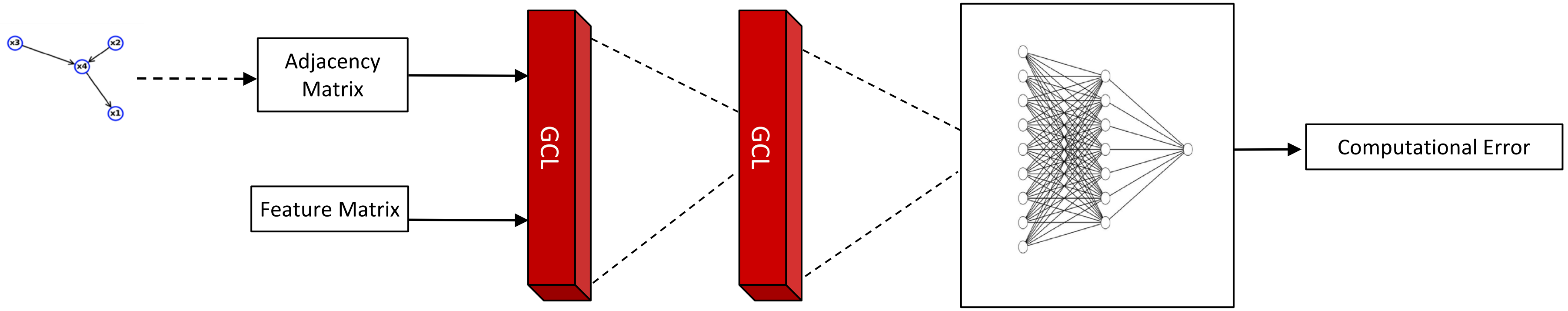
- E.g, if $x_4 \rightarrow x_1$, granting a larger number of bits to represent x_4 would be pointless since the final precision is governed by the precision of the result variable x_1
  - Less truncation and approximation, therefore a reduced error associated with the configuration
  - In practice, configurations where $x_4 \leq x_1$ are associated with smaller errors

- This information can be added to the training set as a collection of additional features
  - If $x_j \rightarrow x_i$ we can create a new feature $F_{i,j} = x_j - x_i$
  - $F_{i,j}$ is added to the dataset
  - Each feature corresponds to one of the logic binary constraints used to express the domain knowledge



2. Ad-hoc Network Topology

- Supervised regression problem whose prior information can be expressed through a graph dependency graph
- We used a spectral graph convolution neural network, implemented via Graph Convolutional Layers (GCL)
  - The input is defined merging the adjacency matrix representing the graph of dependencies and the feature matrix
  - The GCL output is passed to a series of dense layers with decreasing width



3. Regularization & Data augmentation

- We can enforce a monotonicity constraint through a regularization approach

- The loss function is revised adding a penalty term:

$$MSE(X, y) + \lambda \sum_{i,j \in P} max(0, f(x_j) - f(x_i))$$

where assume $x_j \succ x_i$, and λ weight for the regularization term

- Optimize the multipliers of the regularization term [*Fioretto et al., Lagrangian Duality for Constraint Deep Learning, ECML PKKD 2020*]

- We can create new observations for the regularization term exploiting the dominance relation between configurations – Data augmentation
  - Given an instance of the training set, we can easily create a configuration which have higher or lower levels of precision