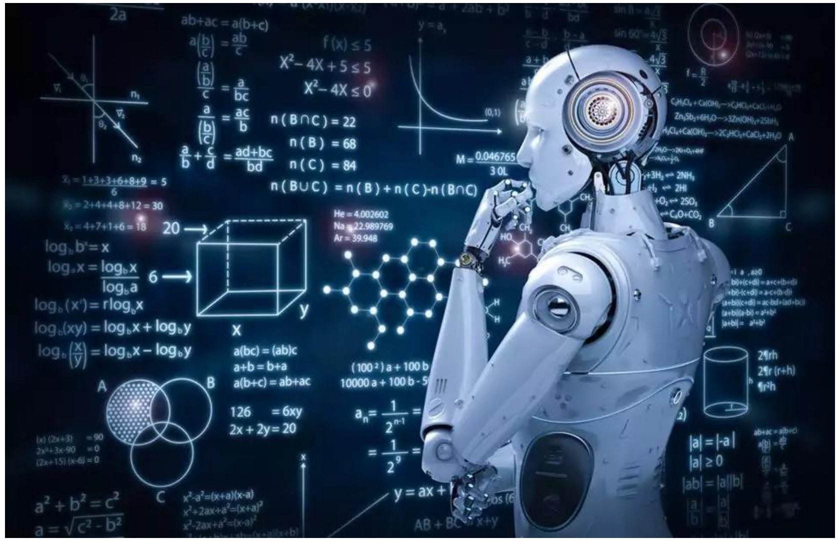
Our contribution

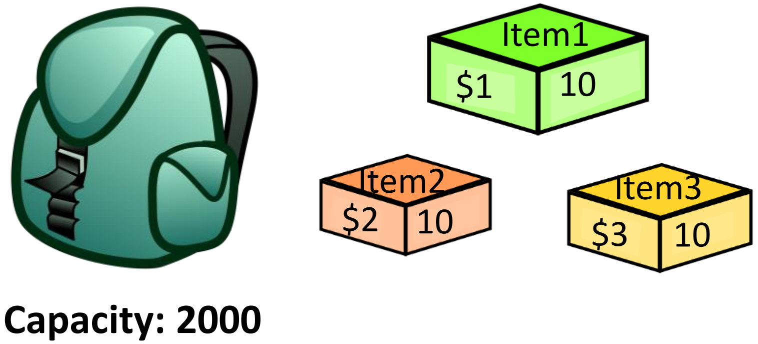
Correctional Regret for Predict+Optimize with Unknown Objectives and Constraints

[Demirovı́ c et al., 2019a], [Elmachtoub and Grigas, 2022]

Work in progress -- IJCAI 2022 DSO Workshop

Xinyi Hu[1] , Jasper C.H. Lee[2,3] , Jimmy H.M. Lee[1] and Allen Z. Zhong[1]
[1]Department of Computer Science and Engineering,
The Chinese University of Hong Kong, Shatin, N.T., Hong Kong
[2]Department of Computer Sciences, University of Wisconsin–Madison, WI, USA
[3]Institute for Foundations of Data Science, University of Wisconsin–Madison, WI, USA





Machine learning

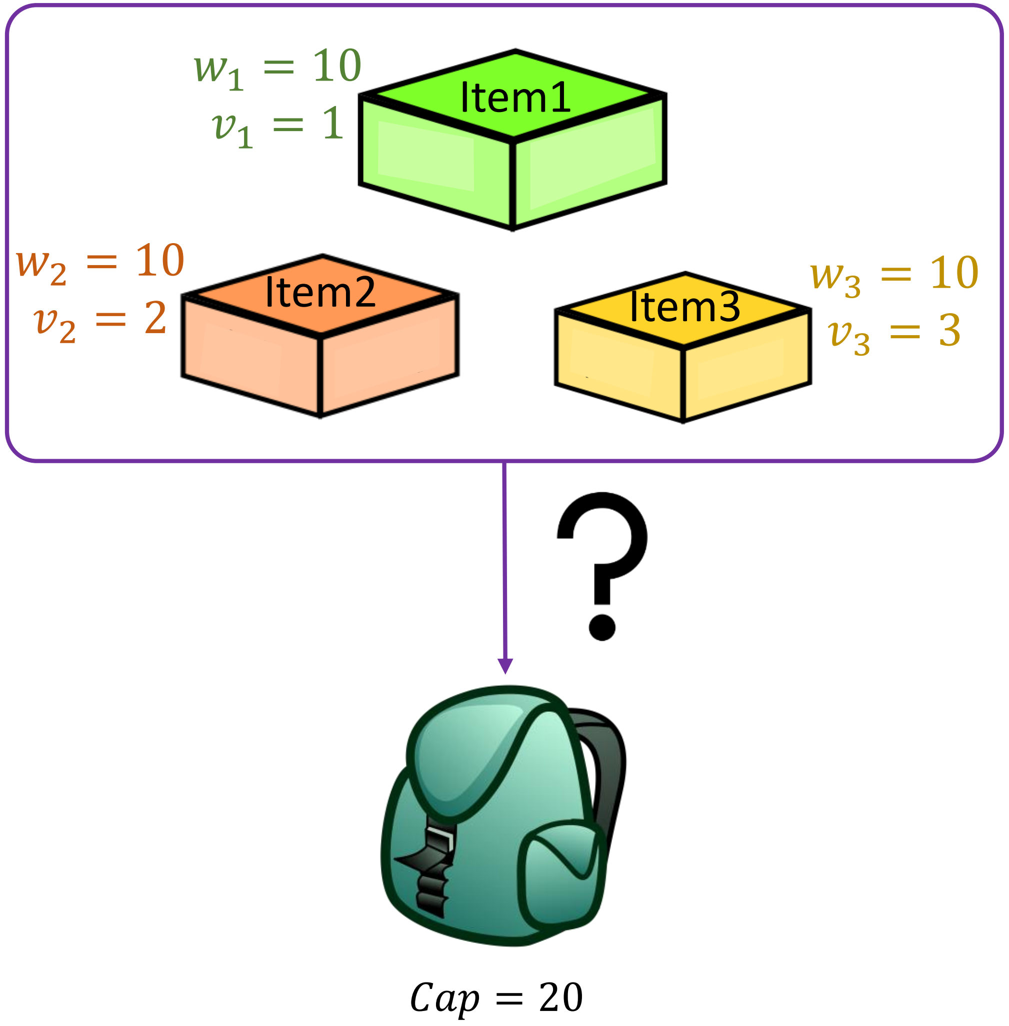
Constraint optimization

Predict+Optimize

Constraint Optimization Problems (COPs)
with unknown parameters

[Demirovĭ c et al., 2019a],
[Demirovĭ c et al., 2019b],
[Elmachtoub and Grigas, 2022]

Knapsack Problem



- 3 items, each with a weight w_i and a value v_i, the capacity Cap is limited.
- Select items so that
  - the total weight is no more than the capacity and
  - maximize the total value
- **Constraint Optimization Problem (COP):**

Decision variable

$$x_i = \begin{cases} 0, & \text{the } i\text{th item is not selected} \\ 1, & \text{the } i\text{th item is selected} \end{cases}$$

Objective function

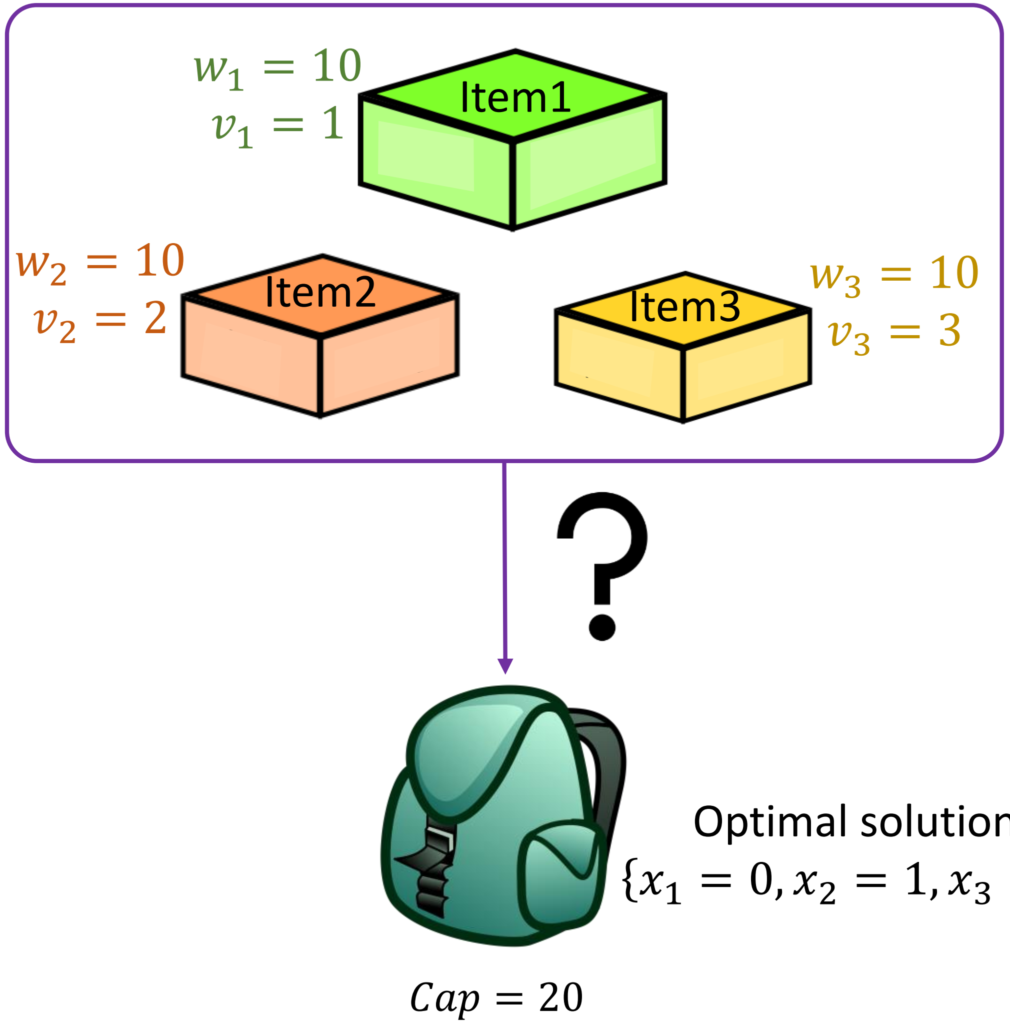
$$\boxed{\max_x x_1 + 2x_2 + 3x_3}$$

Constraints

$$\boxed{\begin{aligned} s.t.\, 10x_1 + 10x_2 + 10x_3 \leq 20 \\ x_i \in \{0,1\}\ \forall i \in \{1,2,3\} \end{aligned}}$$

A constraint is a condition of an optimization problem that the solution must satisfy.

Knapsack Problem



Optimal solution:
$\{x_1 = 0, x_2 = 1, x_3 = 1\}$

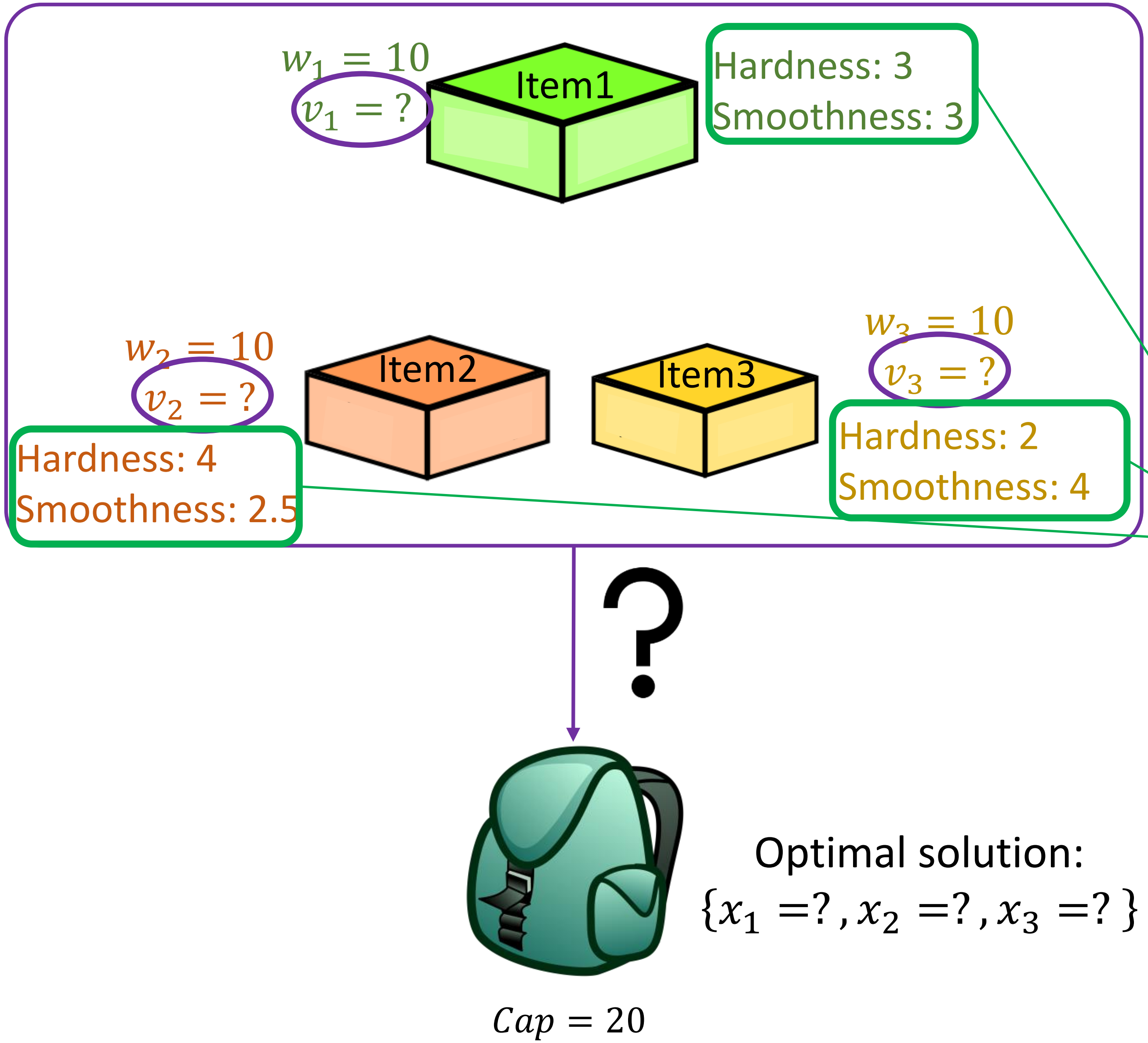
- 3 items, each with a weight w_i and a value v_i, the capacity Cap is limited.
- Select items so that
  - the total weight is no more than the capacity and
  - maximize the total value
- **Constraint Optimization Problem (COP):**

$$\max_x \sum_i v_i x_i$$
$$s.t. \sum_i w_i x_i \le Cap$$
$$x_i \in \{0,1\} \; \forall i \in \{1,2,3\}$$

Problem parameters

NP-hard

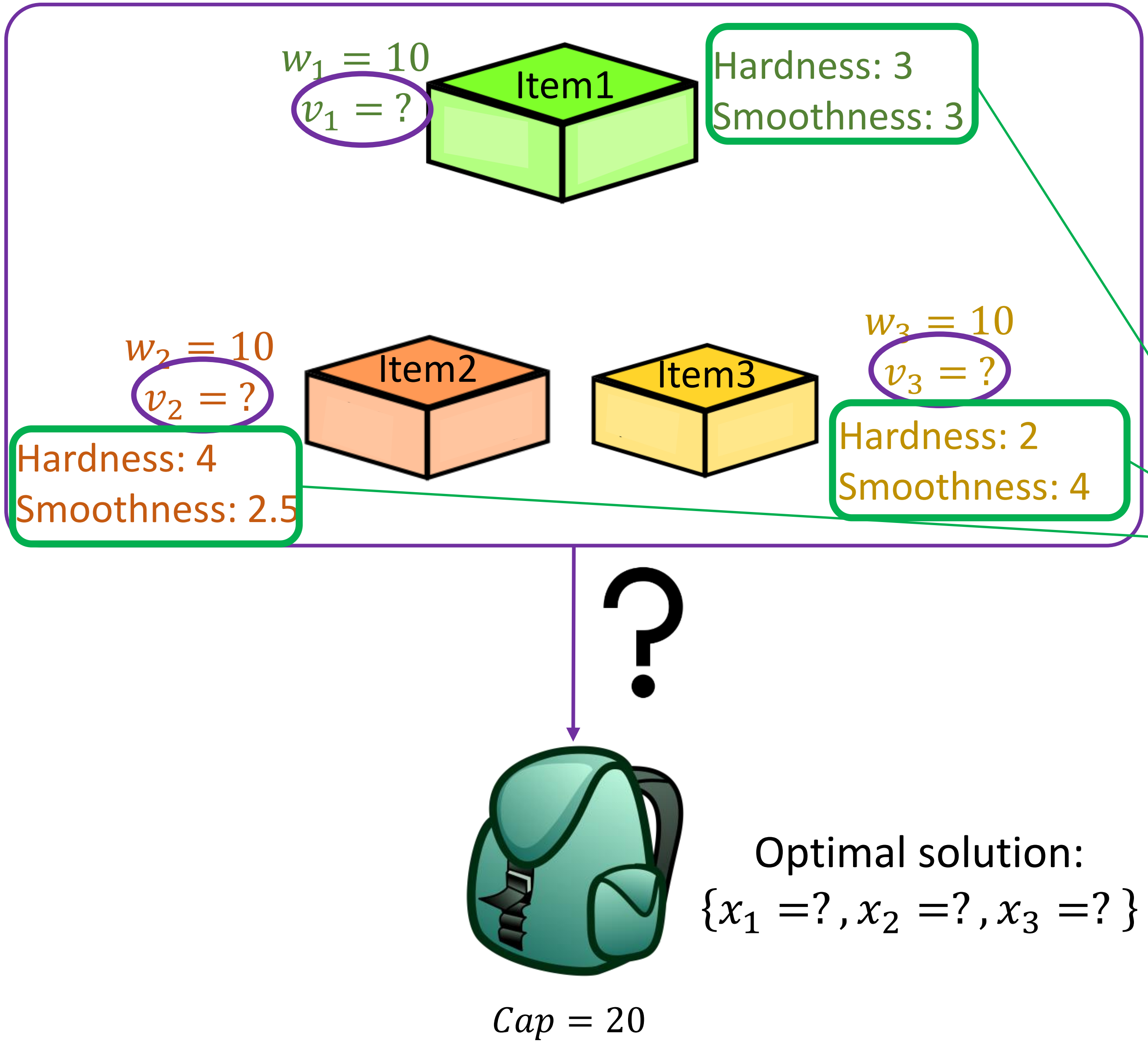
Some problem parameters may be unknown



- The value v_i is unknown.
- Select items so that
  - the total weight is no more than the capacity and
  - maximize the total value

- **COP with Unknown Parameters:**
  - θ: unknown parameters, e.g., $\theta = \{v_1, v_2, v_3\}$
  - A: feature matrix
    - Hardness
    - Smoothness
    - $A = \begin{bmatrix} 3 & 3 \\ 4 & 2.5 \\ 2 & 4 \end{bmatrix}$

Some problem parameters may be unknown

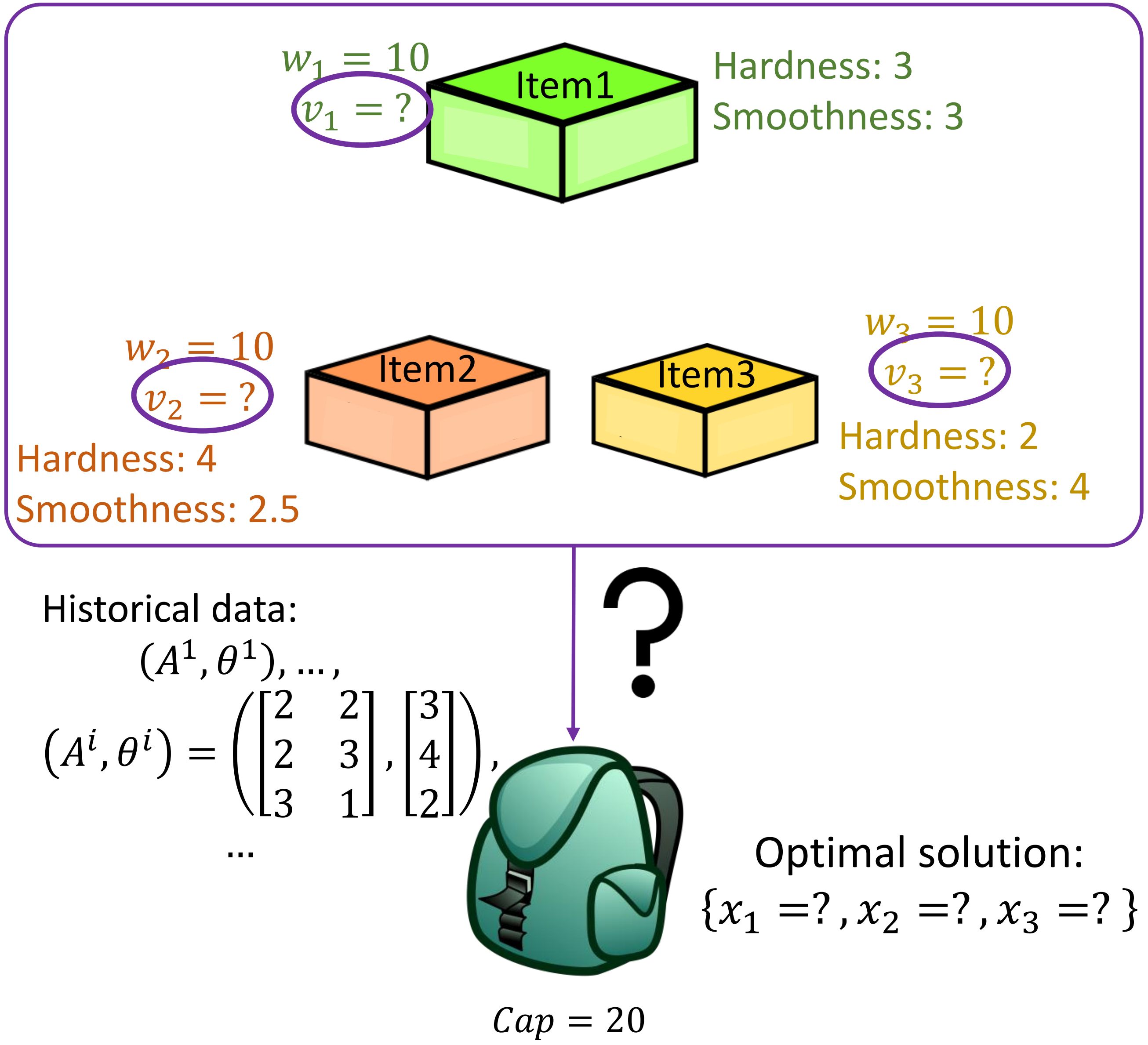


- The value v_i is unknown.
- Select items so that
  - the total weight is no more than the capacity and
  - maximize the total value

- **COP with Unknown Parameters:**
  - θ: unknown parameters, e.g., $\theta = \{v_1, v_2, v_3\}$
  - A: feature matrix
    - Hardness
    - Smoothness
    - $A = \begin{bmatrix} 3 & 3 \\ 4 & 2.5 \\ 2 & 4 \end{bmatrix}$
  - Historical data: $\{(A^1, \theta^1), (A^2, \theta^2), \dots, (A^k, \theta^k)\}$

$$(A^i, \theta^i) = \left(\overbrace{\begin{bmatrix} 2 & 2 \\ 2 & 3 \\ 3 & 1 \end{bmatrix}}^{\text{Historical features}}, \overbrace{\begin{bmatrix} 3 \\ 4 \\ 2 \end{bmatrix}}^{\text{True parameters}} \right)$$

Knapsack Problem



$w_1 = 10$

$v_1 = ?$

Item1

Hardness: 3

Smoothness: 3

$w_2 = 10$

$v_2 = ?$

Hardness: 4

Smoothness: 2.5

Item2

Item3

$w_3 = 10$

$v_3 = ?$

Hardness: 2

Smoothness: 4

Historical data:

$(A^1, \theta^1), \dots,$

$$(A^i, \theta^i) = \left(\begin{bmatrix} 2 & 2 \\ 2 & 3 \\ 3 & 1 \end{bmatrix}, \begin{bmatrix} 3 \\ 4 \\ 2 \end{bmatrix} \right),$$

…

Optimal solution:

$\{x_1 = ?, x_2 = ?, x_3 = ?\}$

$Cap = 20$

- Constraint Optimization Problem (COP):

$$\max_x \sum_i v_i x_i$$
$$s.t. \sum_i w_i x_i \le Cap$$
$$x_i \in \{0,1\} \ \forall i \in \{1,2,3\}$$

- COP with Unknown Parameters:

Unknown parameters

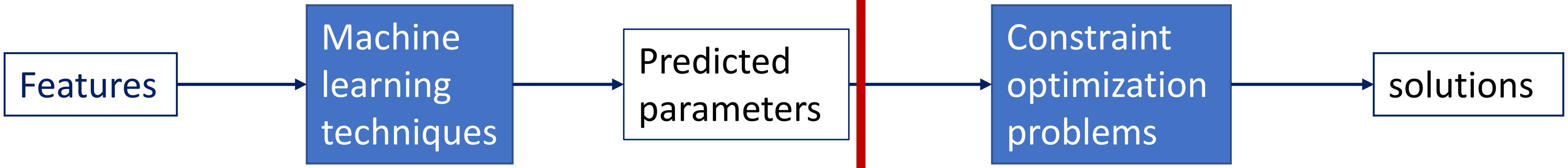
$$\max_x \sum_i \theta_i x_i$$
$$s.t. \sum_i w_i x_i \le Cap$$
$$x_i \in \{0,1\} \ \forall i \in \{1,2,3\}$$

- Aim:
  - learn a prediction function f
  - given current features, use f to generate predicted parameters $\hat{\theta}$
  - try to estimate optimal solution(s) of the COP by using $\hat{\theta}$

How to solve the problem

Predict

Optimize



Features

Machine learning techniques

Predicted parameters

Constraint optimization problems

solutions

Predict+Optimize VS Classical approaches
Main difference:
error measurement

[Demirovı́ c et al., 2020],
[Elmachtoub and Grigas, 2022],
[Guler et al., 2022]

Aim: good estimated solutions of the COP under the true parameters

Classical approaches: predict then optimize

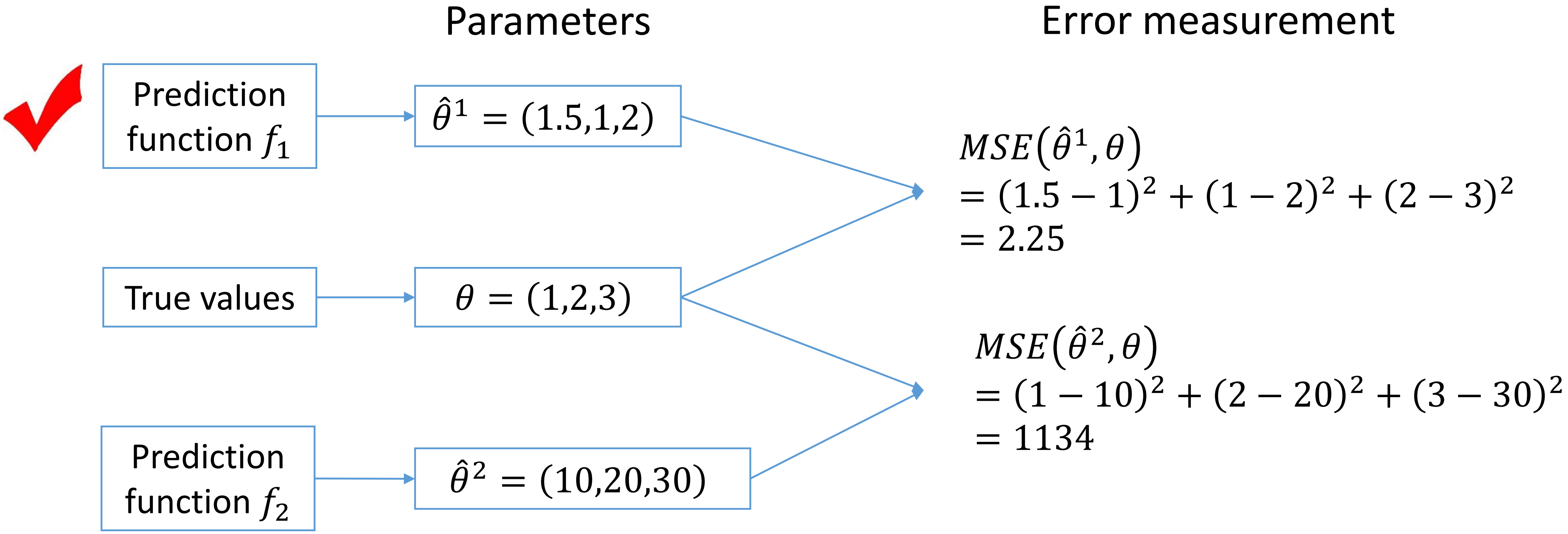
2 separated stage approach:

- Predict: Use standard machine learning techniques to estimate parameters independently of the COP;
  - Training: find a good prediction function that can make best forecast
- Optimize: Use these estimated parameters to solve the COP

Classical approaches: predict then optimize

2 separated stage approach:

- Predict: Use standard machine learning techniques to estimate parameters independently of the COP;
- Optimize: Use these estimated parameters to solve the COP



Parameters

Error measurement

✓ Prediction function f_1 → $\hat{\theta}^1 = (1.5,1,2)$

True values → $\theta = (1,2,3)$

Prediction function f_2 → $\hat{\theta}^2 = (10,20,30)$

$$MSE(\hat{\theta}^1, \theta)$$
$$= (1.5-1)^2 + (1-2)^2 + (2-3)^2$$
$$= 2.25$$

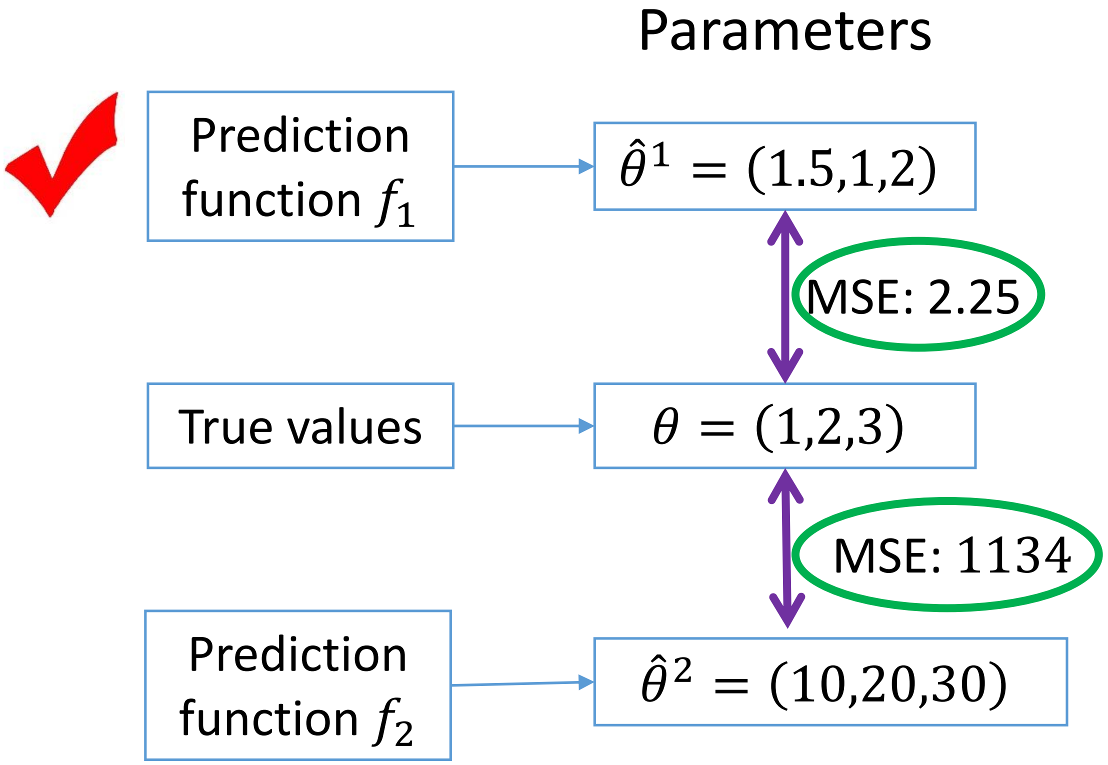
$$MSE(\hat{\theta}^2, \theta)$$
$$= (1-10)^2 + (2-20)^2 + (3-30)^2$$
$$= 1134$$

Classical approaches: predict then optimize

2 separated approach:

Predict: Use standard machine learning techniques to estimate parameters independently of the COP;

Optimize: Use these estimated parameters to solve the COP



Parameters
✓
Prediction function f_1
$\hat{\theta}^1 = (1.5,1,2)$
MSE: 2.25
True values
$\theta = (1,2,3)$
MSE: 1134
Prediction function f_2
$\hat{\theta}^2 = (10,20,30)$

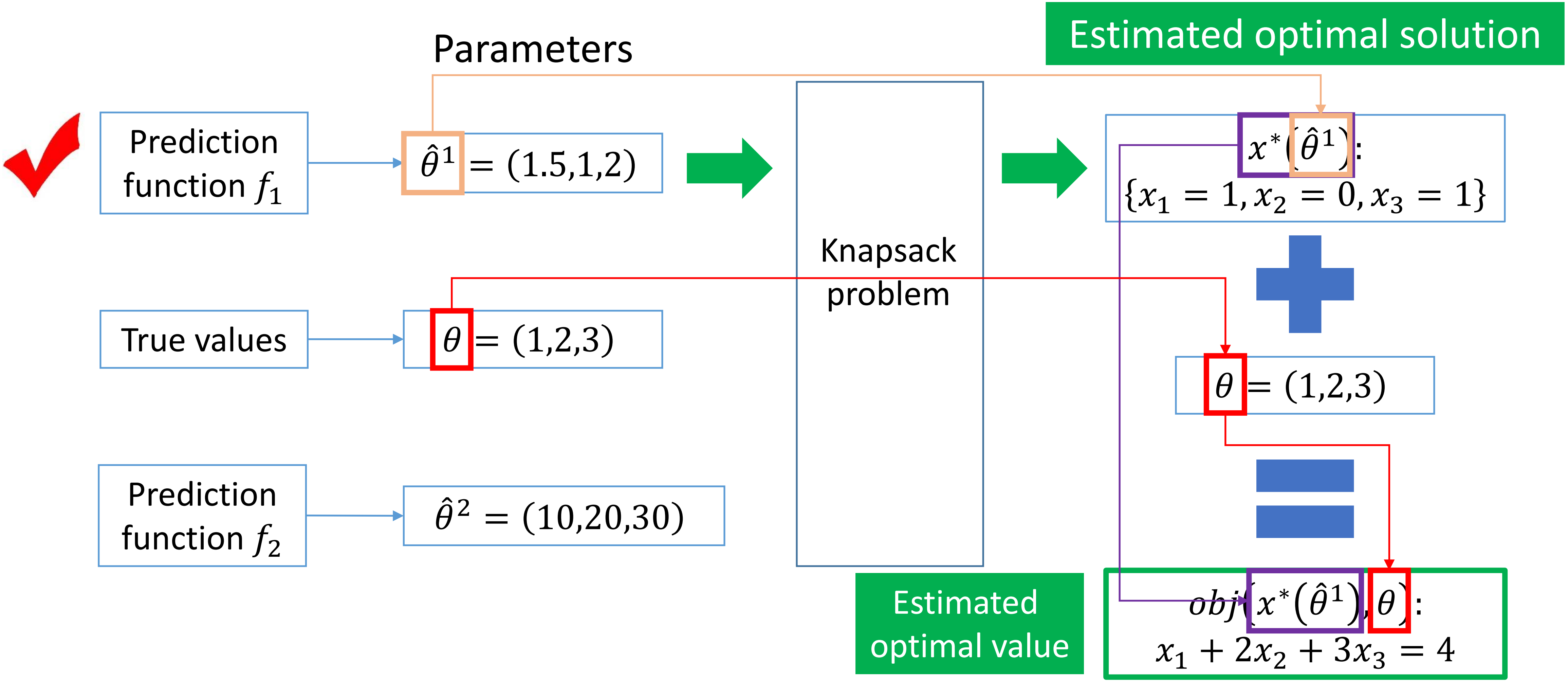
The prediction part is independent of the COP.

Classical approaches aim at minimizing the difference between estimated parameters values and true parameters values.

→ Prediction function f_1 is better.

However…

the best forecast may have a poor result when employed in the COP



Parameters

Estimated optimal solution

✓ Prediction function f_1 → $\hat{\theta}^1 = (1.5,1,2)$

True values → $\theta = (1,2,3)$

Prediction function f_2 → $\hat{\theta}^2 = (10,20,30)$

Knapsack problem

$x^*(\hat{\theta}^1)$:
$\{x_1 = 1, x_2 = 0, x_3 = 1\}$

+

$\theta = (1,2,3)$

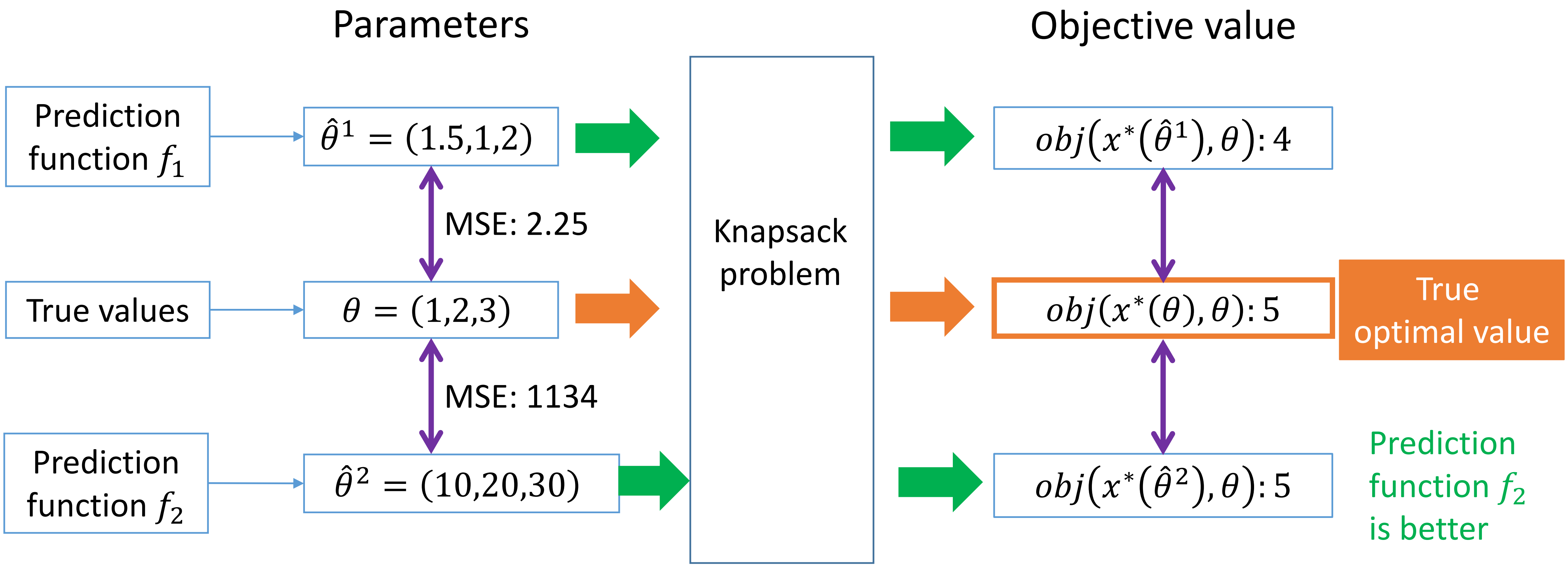
=

$obj(x^*(\hat{\theta}^1), \theta)$:
$x_1 + 2x_2 + 3x_3 = 4$

Estimated optimal value

However…

the best forecast may have a poor result when employed in the COP



Parameters

Objective value

Prediction function f_1

$\hat{\theta}^1 = (1.5,1,2)$

MSE: 2.25

True values

$\theta = (1,2,3)$

MSE: 1134

Prediction function f_2

$\hat{\theta}^2 = (10,20,30)$

Knapsack problem

$obj\left(x^*\left(\hat{\theta}^1\right), \theta\right): 4$

$obj\left(x^*(\theta), \theta\right): 5$

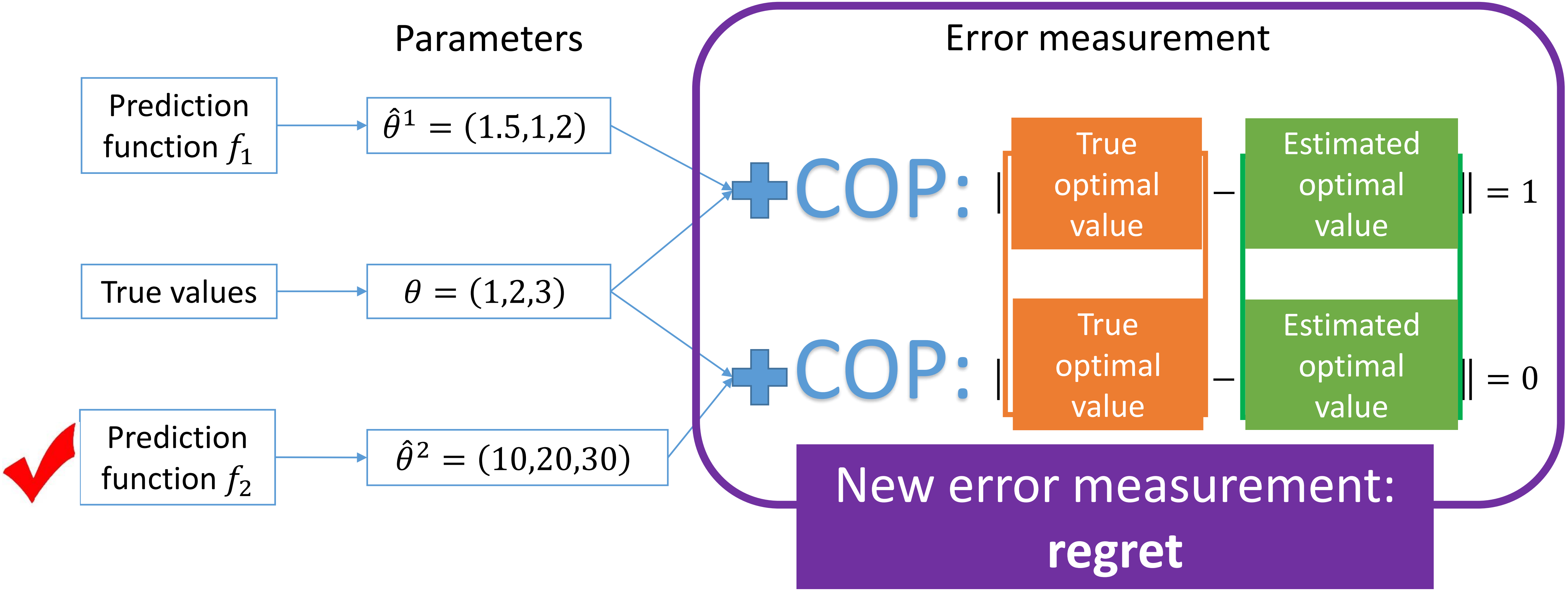
True optimal value

$obj\left(x^*\left(\hat{\theta}^2\right), \theta\right): 5$

Prediction function f_2 is better

Predict+Optimize

Take the COP into account when doing the prediction



Parameters

Error measurement

Prediction function f_1 → $\hat{\theta}^1 = (1.5,1,2)$

True values → $\theta = (1,2,3)$

Prediction function f_2 → $\hat{\theta}^2 = (10,20,30)$

+COP: $\|$ True optimal value $-$ Estimated optimal value $\| = 1$

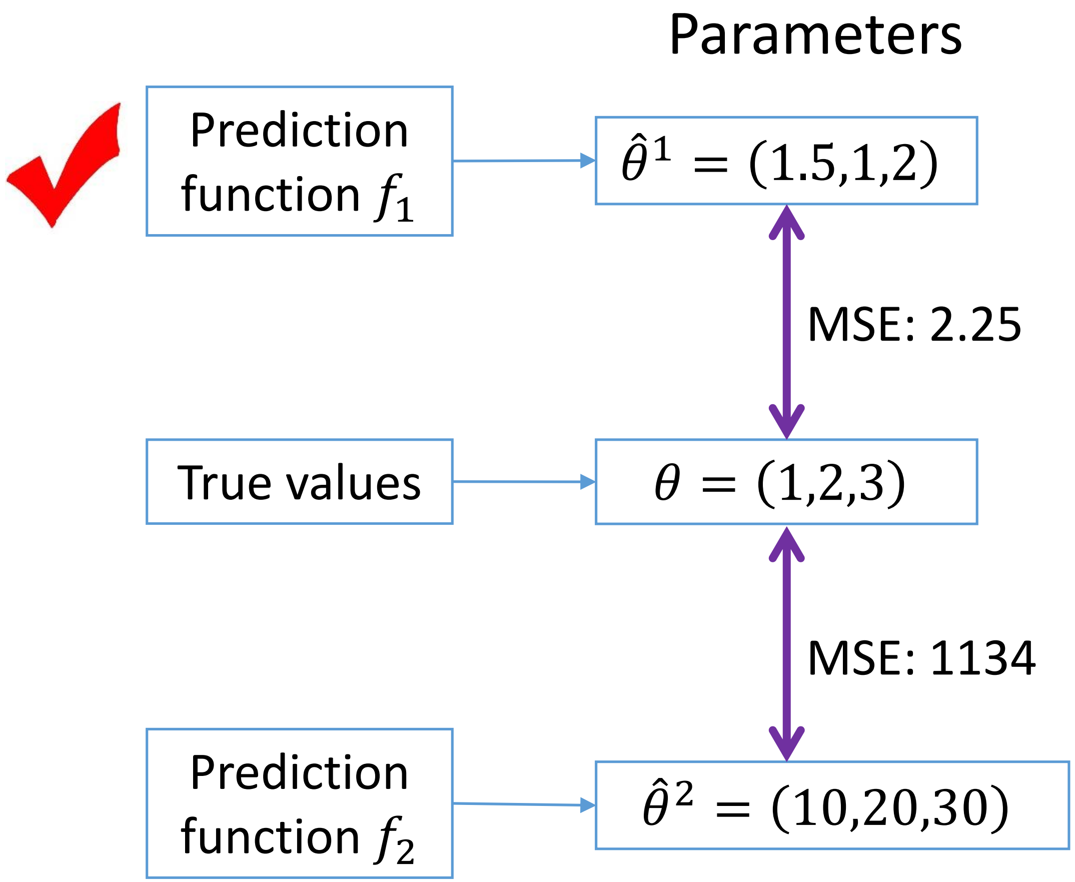
+COP: $\|$ True optimal value $-$ Estimated optimal value $\| = 0$

New error measurement: **regret**

[Demiroví c et al., 2019a], [Demiroví c et al., 2019b], [Guler et al., 2022]

Comparison

Classical approaches VS Predict+Optimize



Parameters

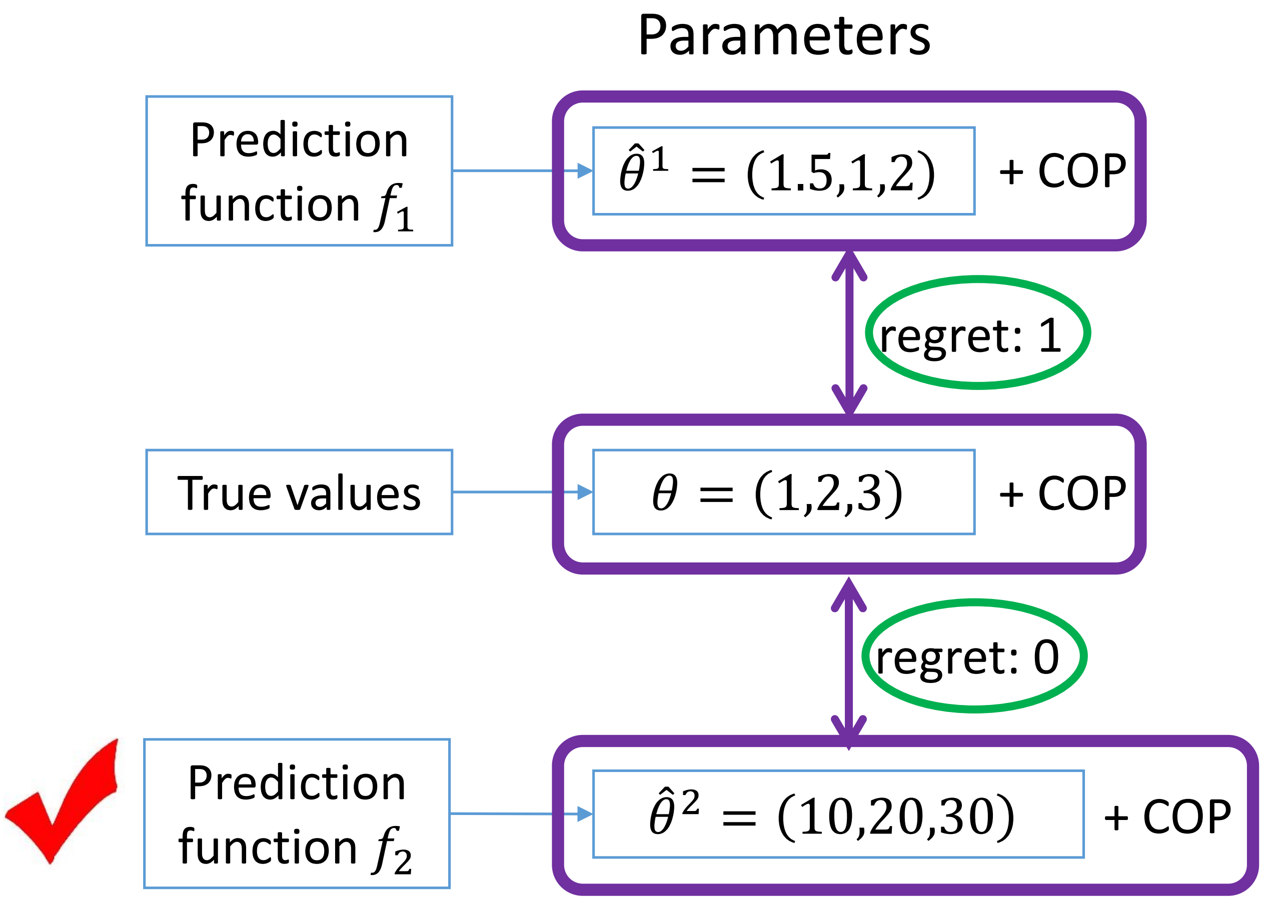
Prediction function f_1 → $\hat{\theta}^1 = (1.5,1,2)$

MSE: 2.25

True values → $\theta = (1,2,3)$

MSE: 1134

Prediction function f_2 → $\hat{\theta}^2 = (10,20,30)$



Parameters

Prediction function f_1 → $\hat{\theta}^1 = (1.5,1,2)$ + COP

regret: 1

True values → $\theta = (1,2,3)$ + COP

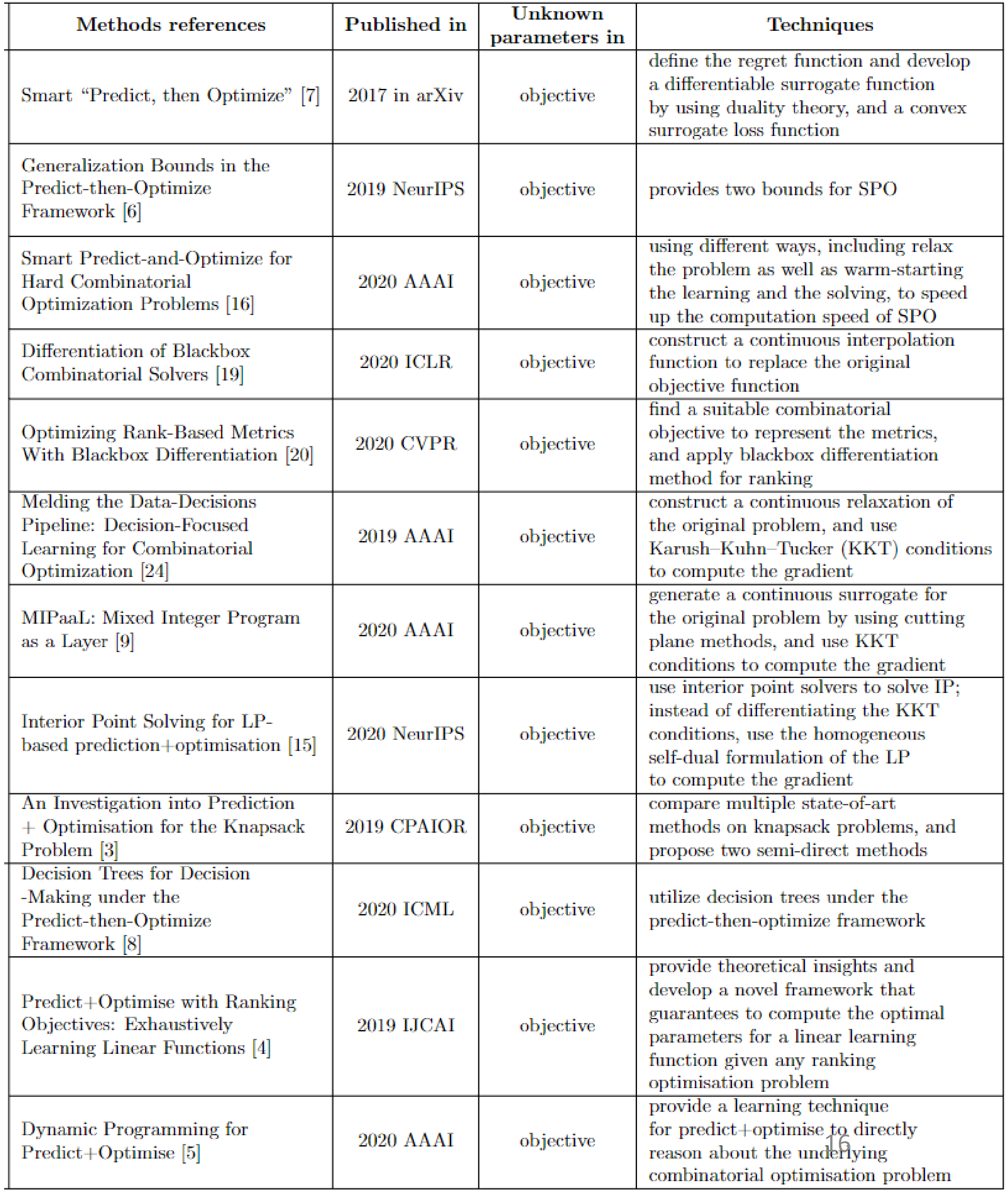
regret: 0

Prediction function f_2 → $\hat{\theta}^2 = (10,20,30)$ + COP

[Demirovi ́ c et al., 2019a], [Demirovi ́ c et al., 2019b], [Guler et al., 2022]

Related Works

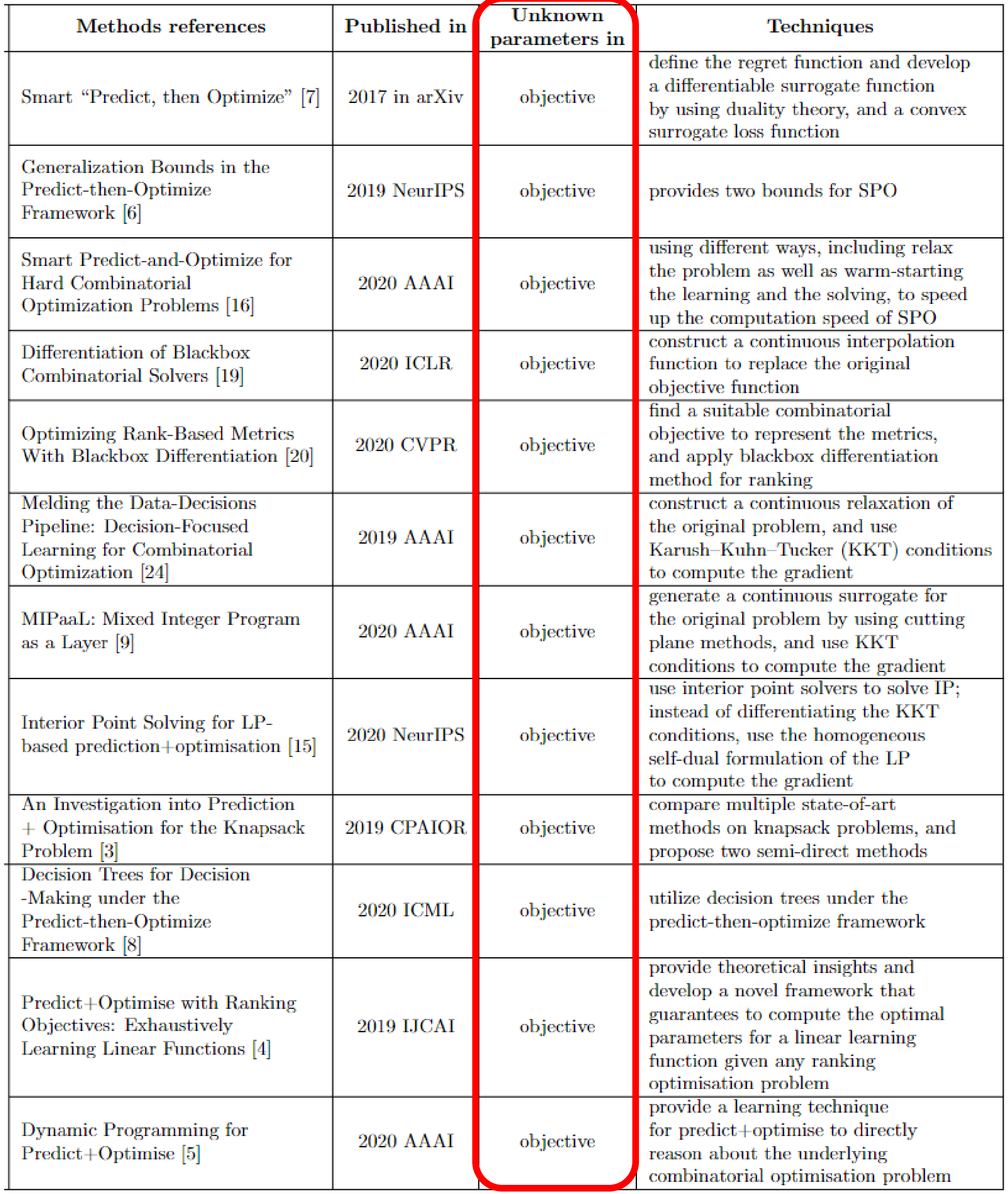
- The regret function is non-differentiable, which is unfriendly to any gradient-based learning process
- All of the related works focus on how to overcome the non-differentiability and train with the new loss.



Methods references	Published in	Unknown parameters in	Techniques
Smart "Predict, then Optimize" [7]	2017 in arXiv	objective	define the regret function and develop a differentiable surrogate function by using duality theory, and a convex surrogate loss function
Generalization Bounds in the Predict-then-Optimize Framework [6]	2019 NeurIPS	objective	provides two bounds for SPO
Smart Predict-and-Optimize for Hard Combinatorial Optimization Problems [16]	2020 AAAI	objective	using different ways, including relax the problem as well as warm-starting the learning and the solving, to speed up the computation speed of SPO
Differentiation of Blackbox Combinatorial Solvers [19]	2020 ICLR	objective	construct a continuous interpolation function to replace the original objective function
Optimizing Rank-Based Metrics With Blackbox Differentiation [20]	2020 CVPR	objective	find a suitable combinatorial objective to represent the metrics, and apply blackbox differentiation method for ranking
Melding the Data-Decisions Pipeline: Decision-Focused Learning for Combinatorial Optimization [24]	2019 AAAI	objective	construct a continuous relaxation of the original problem, and use Karush–Kuhn–Tucker (KKT) conditions to compute the gradient
MIPaaL: Mixed Integer Program as a Layer [9]	2020 AAAI	objective	generate a continuous surrogate for the original problem by using cutting plane methods, and use KKT conditions to compute the gradient
Interior Point Solving for LP-based prediction+optimisation [15]	2020 NeurIPS	objective	use interior point solvers to solve IP; instead of differentiating the KKT conditions, use the homogeneous self-dual formulation of the LP to compute the gradient
An Investigation into Prediction + Optimisation for the Knapsack Problem [3]	2019 CPAIOR	objective	compare multiple state-of-art methods on knapsack problems, and propose two semi-direct methods
Decision Trees for Decision -Making under the Predict-then-Optimize Framework [8]	2020 ICML	objective	utilize decision trees under the predict-then-optimize framework
Predict+Optimise with Ranking Objectives: Exhaustively Learning Linear Functions [4]	2019 IJCAI	objective	provide theoretical insights and develop a novel framework that guarantees to compute the optimal parameters for a linear learning function given any ranking optimisation problem
Dynamic Programming for Predict+Optimise [5]	2020 AAAI	objective	provide a learning technique for predict+optimise to directly reason about the underlying combinatorial optimisation problem

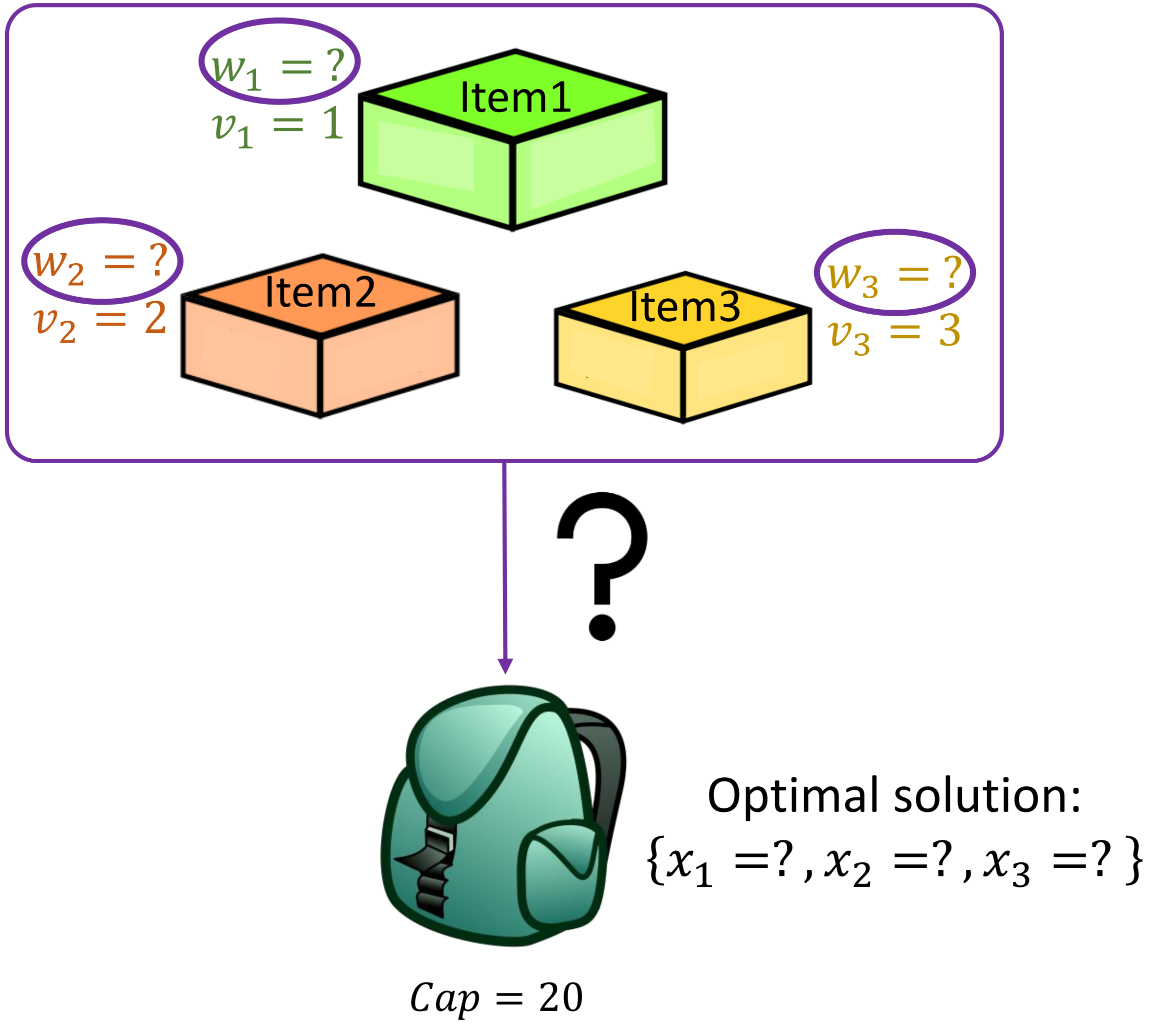
Related Works

What if the constraints also contain unknown parameters?



Methods references	Published in	Unknown parameters in	Techniques
Smart "Predict, then Optimize" [7]	2017 in arXiv	objective	define the regret function and develop a differentiable surrogate function by using duality theory, and a convex surrogate loss function
Generalization Bounds in the Predict-then-Optimize Framework [6]	2019 NeurIPS	objective	provides two bounds for SPO
Smart Predict-and-Optimize for Hard Combinatorial Optimization Problems [16]	2020 AAAI	objective	using different ways, including relax the problem as well as warm-starting the learning and the solving, to speed up the computation speed of SPO
Differentiation of Blackbox Combinatorial Solvers [19]	2020 ICLR	objective	construct a continuous interpolation function to replace the original objective function
Optimizing Rank-Based Metrics With Blackbox Differentiation [20]	2020 CVPR	objective	find a suitable combinatorial objective to represent the metrics, and apply blackbox differentiation method for ranking
Melding the Data-Decisions Pipeline: Decision-Focused Learning for Combinatorial Optimization [24]	2019 AAAI	objective	construct a continuous relaxation of the original problem, and use Karush–Kuhn–Tucker (KKT) conditions to compute the gradient
MIPaaL: Mixed Integer Program as a Layer [9]	2020 AAAI	objective	generate a continuous surrogate for the original problem by using cutting plane methods, and use KKT conditions to compute the gradient
Interior Point Solving for LP-based prediction+optimisation [15]	2020 NeurIPS	objective	use interior point solvers to solve IP; instead of differentiating the KKT conditions, use the homogeneous self-dual formulation of the LP to compute the gradient
An Investigation into Prediction + Optimisation for the Knapsack Problem [3]	2019 CPAIOR	objective	compare multiple state-of-art methods on knapsack problems, and propose two semi-direct methods
Decision Trees for Decision -Making under the Predict-then-Optimize Framework [8]	2020 ICML	objective	utilize decision trees under the predict-then-optimize framework
Predict+Optimise with Ranking Objectives: Exhaustively Learning Linear Functions [4]	2019 IJCAI	objective	provide theoretical insights and develop a novel framework that guarantees to compute the optimal parameters for a linear learning function given any ranking optimisation problem
Dynamic Programming for Predict+Optimise [5]	2020 AAAI	objective	provide a learning technique for predict+optimise to directly reason about the underlying combinatorial optimisation problem

If the constraints contain unknown parameters



Optimal solution:
$\{x_1 = ?, x_2 = ?, x_3 = ?\}$

$Cap = 20$

- Knapsack with unknown weights

$$\max_x \sum_i v_i x_i$$
$$s.t. \sum_i \theta_i x_i \le Cap$$
$$x_i \in \{0,1\} \; \forall i \in \{1,2,3\}$$

Unknown parameters in constraints

- Eg.

Estimated weights: $\{\widehat{w_1} = \widehat{w_2} = \widehat{w_3} = 5\}$

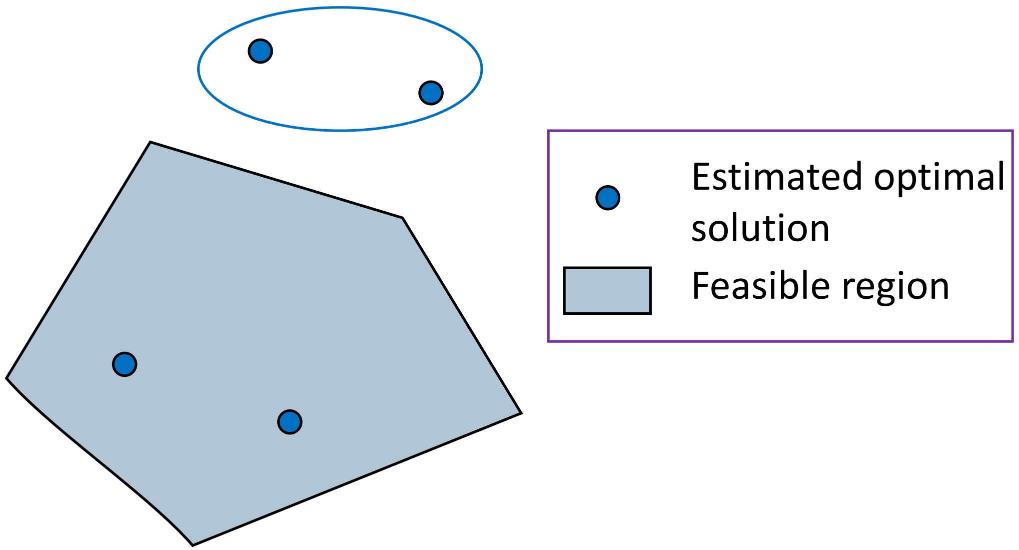
Estimated optimal solution:
$\{x_1 = x_2 = x_3 = 1\}$
(True weights: $\{w_1 = w_2 = w_3 = 10\}$)

infeasible

- The estimated optimal solution may be infeasible under the true parameters

Regret is inapplicable

- Unknown parameters appearing in constraints **(more complex)**
  - the estimated optimal solution may be out of the true solution space



- Regret: does not take feasibility into account

$$Regret(\hat{\theta},\theta) = \| \text{True optimal value} - \text{Estimated optimal value} \|$$

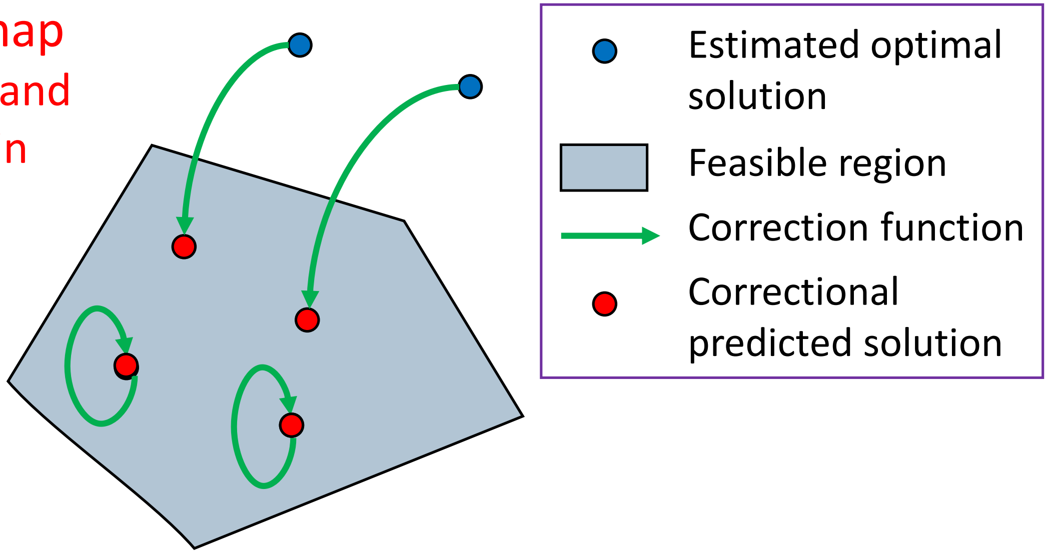
inapplicable

Our Work: Correction Function

Some applications:

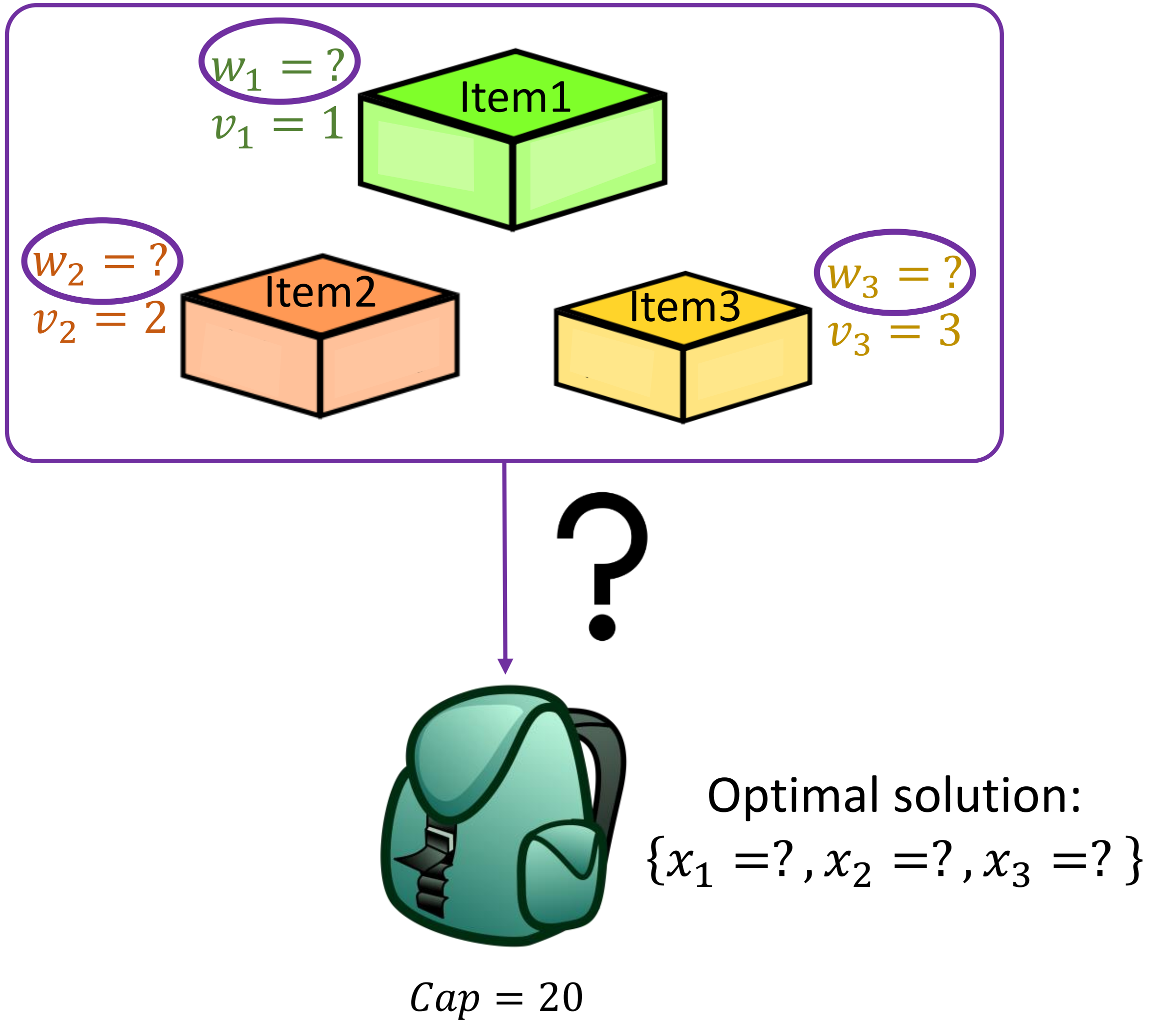
allow solution modification after true parameters are revealed

- **correction function** should map
  - (a) every feasible solution to itself and
  - (b) each infeasible solution to one in the feasible region



The space of possible correction functions : problem and application specific

Case Study 1: Knapsack



- If the weights are unknown?

$$\max_{x} \sum_{i} v_i x_i$$
$$s.t. \sum_i \theta_i x_i \leq Cap$$
$$x_i \in \{0,1\} \; \forall i \in \{1,2,3\}$$

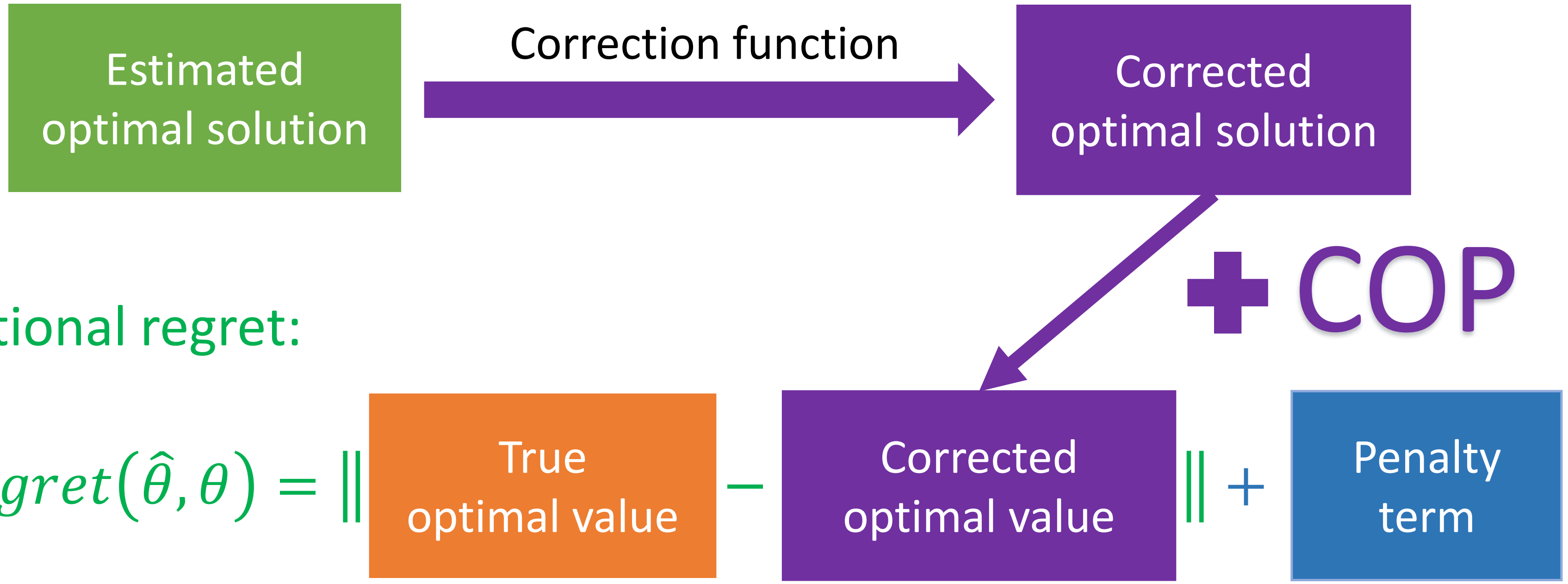
- When the total weight of the selected items exceeds the capacity:
  - Correction function 1: remove all items
  - Correction function 2: remove the items one by one in increasing order of values

Our Work: Correctional Regret

To cater for unknown parameters appearing in constraints

- Correction function:

- Correctional regret:



$CRegret(\hat{\theta}, \theta) = \|$ True optimal value $-$ Corrected optimal value $\| +$ Penalty term

E.g. knapsack problem with unknown weights: removal fee

Experiment Setting

Comparison algorithms

proposed

Comparison algorithms	Branch and learn (B&L) [Hu et al., 2022]	Branch and learn with correction (B&L-C)	Linear regression (LR)	k-nearest neighbors (k-NN)	Classification and regression tree (CART)	Random forest (RF)
Category	Predict+Optimize method	Extension of B&L	Classical regression methods			
Trained by	Regret	Correctional regret	Mean square error (MSE)			
Tested by	Correctional regret					

Experiment Setting

Maximum flow

- Unknown parameters in constraints
- 2 Real-life graphs
  - USANet, 24 vertices and 43 edges
  - GEANT, 40 vertices and 61 edges
- Artificial and real-life datasets

Minimum cost vertex cover

- Unknown parameters in **both** the objective and constraints
- 2 Real-life graphs
  - ABILENE, 12 vertices and 15 edges
  - GEANT, 11 vertices and 34 edges
- Artificial and real-life datasets

Experiment Dataset

Real life dataset

- ICON energy-aware scheduling competition
- Also used in previous works on Predict+Optimize
- Each parameter has 8 features

Artificial dataset

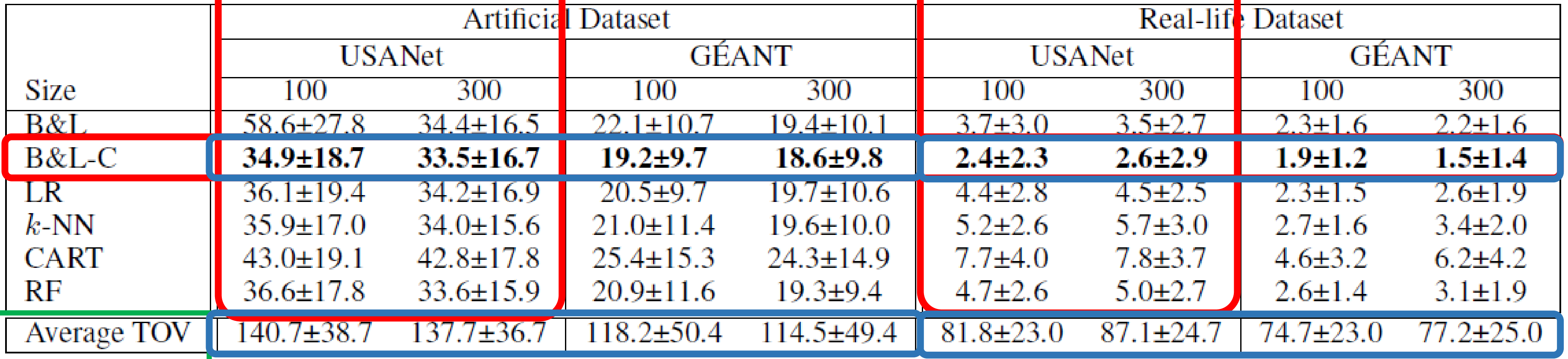
- $100 * \sin(a_1) * \sin(a_2) + 10 * \sin(a_3) * \sin(a_4)$

Highly nonlinear

Experiment Results: Maximum Flow

≥ 0.3% smaller correctional regret

≥ 25% smaller correctional regret



	Artificial Dataset				Real-life Dataset			
	USANet		GÉANT		USANet		GÉANT	
Size	100	300	100	300	100	300	100	300
B&L	58.6±27.8	34.4±16.5	22.1±10.7	19.4±10.1	3.7±3.0	3.5±2.7	2.3±1.6	2.2±1.6
B&L-C	**34.9±18.7**	**33.5±16.7**	**19.2±9.7**	**18.6±9.8**	**2.4±2.3**	**2.6±2.9**	**1.9±1.2**	**1.5±1.4**
LR	36.1±19.4	34.2±16.9	20.5±9.7	19.7±10.6	4.4±2.8	4.5±2.5	2.3±1.5	2.6±1.9
k-NN	35.9±17.0	34.0±15.6	21.0±11.4	19.6±10.0	5.2±2.6	5.7±3.0	2.7±1.6	3.4±2.0
CART	43.0±19.1	42.8±17.8	25.4±15.3	24.3±14.9	7.7±4.0	7.8±3.7	4.6±3.2	6.2±4.2
RF	36.6±17.8	33.6±15.9	20.9±11.6	19.3±9.4	4.7±2.6	5.0±2.7	2.6±1.4	3.1±1.9
Average TOV	140.7±38.7	137.7±36.7	118.2±50.4	114.5±49.4	81.8±23.0	87.1±24.7	74.7±23.0	77.2±25.0

TOV:
True Optimal Value

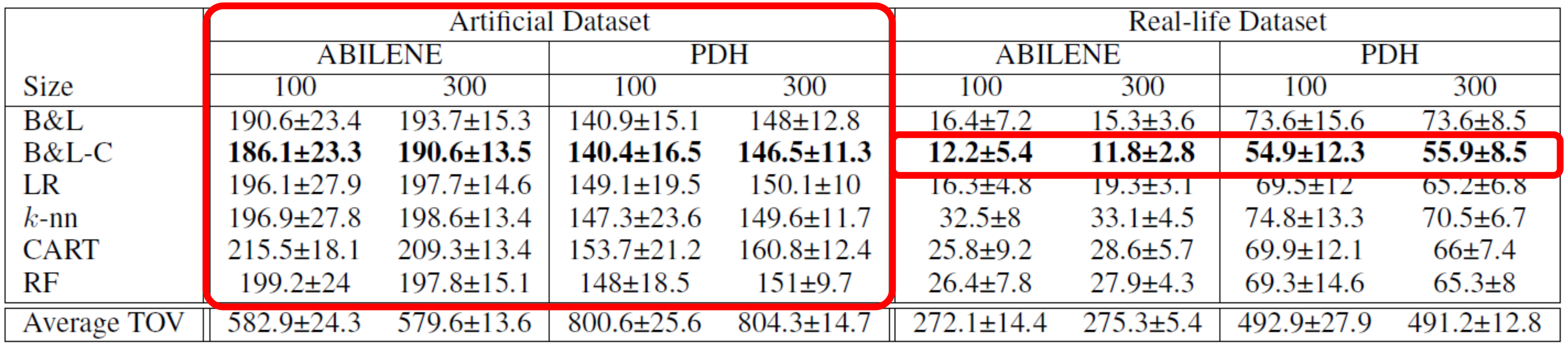
16-24% relative error

2-3% relative error

Table 1: Mean correctional regrets and standard deviations for MFP with unknown capacities.

- B&L-C achieves the best performance in all cases.
- The performance differences among different methods are larger in the real-life dataset, and the advantages of B&L-C are more obvious.
- All methods achieve better performance in the real-life dataset. This is consistent with how the artificial dataset is purposefully designed to be highly non-linear, and thus more difficult to estimate.

Experiment Results: Minimum Cost Vertex Cover



	Artificial Dataset				Real-life Dataset			
	ABILENE		PDH		ABILENE		PDH	
Size	100	300	100	300	100	300	100	300
B&L	190.6±23.4	193.7±15.3	140.9±15.1	148±12.8	16.4±7.2	15.3±3.6	73.6±15.6	73.6±8.5
B&L-C	**186.1±23.3**	**190.6±13.5**	**140.4±16.5**	**146.5±11.3**	**12.2±5.4**	**11.8±2.8**	**54.9±12.3**	**55.9±8.5**
LR	196.1±27.9	197.7±14.6	149.1±19.5	150.1±10	16.3±4.8	19.3±3.1	69.5±12	65.2±6.8
k-nn	196.9±27.8	198.6±13.4	147.3±23.6	149.6±11.7	32.5±8	33.1±4.5	74.8±13.3	70.5±6.7
CART	215.5±18.1	209.3±13.4	153.7±21.2	160.8±12.4	25.8±9.2	28.6±5.7	69.9±12.1	66±7.4
RF	199.2±24	197.8±15.1	148±18.5	151±9.7	26.4±7.8	27.9±4.3	69.3±14.6	65.3±8
Average TOV	582.9±24.3	579.6±13.6	800.6±25.6	804.3±14.7	272.1±14.4	275.3±5.4	492.9±27.9	491.2±12.8

Table 2: Mean correctional regrets and standard deviations for MCVC with unknown costs and edge values.

- B&L-C has the best performance in all cases with the real-life dataset.
- On the artificial dataset, all algorithms perform essentially the same.

Summary

- Predict+Optimize: unknown parameters in objectives + constraints
  - Challenge: estimated solutions may be infeasible
  - Correction function
  - Correctional regret

- Experiment results
  - Maximum flow problem: unknown capacities
  - Minimum cost vertex cover problem: unknown costs + edge values