Introduction

- Resolution of recurrent combinatorial optimization problems, coupling machine learning techniques with branch & bound algorithms, operating under a limited time budget.
- Assuming problems are the realization of a generative process, historical data are collected and used to train a classification model.
- At first, when solving a new instance, this model will select a subset of decision variables to be set heuristically to some reference values, becoming fixed parameters.
- The remaining variables are left free and form a smaller sub-problem whose solution, while being an approximation of the optimum, can be obtained sensibly faster.
- Subsequently, if some of the time allocated is available, an iterative process of blocking/unblocking variables takes place, allowing to explore other areas of the solution space.
- This approach is of particular interest for problems where perturbations on the instance parameters can occur unexpectedly, requiring a rapid re-optimization of a complex model.

Overview & Example

Example: problem $P$ is one of energy production planning. All days are similar, the network is mostly unchanged, only the energy demands vary. recurrent combinatorial optimization.

- An anomaly occurs, some parameters can change: new problem $P'$, re-optimization is necessary.
- Response time is limited: resolution time budget.
- Variations on a theme: generative process.
- Past resolutions, data available.
- Learn from the past, solve future instances faster.

Methods developed:

1. NaïBRec: block irrelevant variables, only one SP generation.
   - Given $p$ variables, SP: smaller sub-problem, manageable approximation.
2. SuSPen: block/unblock variables, iterative SP generation.
   - Given $p$ variables, SP repeatedly, until resolution time is available.

Blocked variables are assigned a value heuristically. E.g., the optimal solution $x^*_P$ to a reference problem $P_{ref}$, such as the model under nominal parameters.

Mixed Integer Programming

- A mixed integer programming (MIP) Problem $P$ is:
  $$z^* = \min \{c^T x \mid x \in \chi\}$$
  where $\chi = \{x \in \mathbb{R}^p \times 2^\lambda, \text{s.t. } Ax \leq b\}$.
- We define a sub-problem $SP$ of $P$ as:
  $$z_{SP} = \min \{c^T x_{SP} + \epsilon \mid x_{SP} \in \chi\}$$
  const
- block $\chi$ (set of) variables to a reference value: $x_{SP} = \{x \mid x := x_{ref}\}$.
- $x_{SP} = x_{SP}$, $x_{ref}$ are the remaining free variables.

MIP is a widely adopted formulation for combinatorial optimization problems, solved via Branch & Bound algorithms. Highly performant solvers exist (CPLEX, GUROBI), for reasonably sized instances.

- Intrinsic exponential complexity: won’t go away.

(1) NaïBRec: Naïve Bayes for Recurrent Problems

How to block variables? Frame as a multi-label classification problem, find variables not affected by random events.

Multi-label Classification (MLC)

- Image annotation:
  $$\Rightarrow \{\text{beautiful, mountains, Dolomites, go, holiday, summer, winter}\}$$
- Genetic data: classification of gene functions.
  $$\Rightarrow \{\text{cell growth, cell multiplication, structural function}\}$$
- Labels set $\{\text{decision variables}\}$
  $$\Rightarrow \text{How many labels to include?} \Rightarrow \text{How many variables to block?} \Rightarrow \text{Which labels to choose?} \Rightarrow \text{Which variables to block?}$$

Note: when variables are too many, use clusters and hierarchies of variables as proxy labels.

NaïBRec MLC Algorithm – Cascade of Predictors

Step (1) Size estimation, $m$: number of labels to be predicted.
Step (2) Sequential label prediction, given the size of the target vector:
$$\forall m \Rightarrow y_1 \Rightarrow y_2 \Rightarrow \ldots \Rightarrow y_{m-1} \Rightarrow y_m,$$

- Each step is a naive Bayes classifier.

NaiBRec Meta-Algorithm

Step (1) Collect past data, train classification model.
Step (2) Extract features from current $P'$, e.g., differences $P' \Rightarrow P_{ref} \Rightarrow A(MIP – parameters)$.
Step (3) $N_{NaïBRec}(\Delta(\text{parameters})) \Rightarrow \{x_{SP}, x_{SP}\}$, predict variables to be blocked.
Step (4) Generate SP
Step (5) Run optimization

(2) SuSPen, Supervised Sub-Problem Generation

Extends NaïBRec, introducing the concept blocking/unblocking decision variables. It explores the solution space while handling small problems. It respects the time constraint imposed.

Perspectives

- Considering the block/unblock process as a Markov Decision Process (MDP).
- Reinforcement Learning framework.

Acknowledgments

This research benefited from the support of the “FMIH Program Gaspard Monge in optimization and operations research” and from the support to this program from EDF.

Acknowledgments

Luca Mossina, PhD student
Dept. Complex Systems Engineering, ISAE-Supaero School of Engineering.
University of Toulouse, France.

Introduction

- Resolution of recurrent combinatorial optimization problems, coupling machine learning techniques with branch & bound algorithms, operating under a limited time budget.
- Assuming problems are the realization of a generative process, historical data are collected and used to train a classification model.
- At first, when solving a new instance, this model will select a subset of decision variables to be set heuristically to some reference values, becoming fixed parameters.
- The remaining variables are left free and form a smaller sub-problem whose solution, while being an approximation of the optimum, can be obtained sensibly faster.
- Subsequently, if some of the time allocated is available, an iterative process of blocking/unblocking variables takes place, allowing to explore other areas of the solution space.
- This approach is of particular interest for problems where perturbations on the instance parameters can occur unexpectedly, requiring a rapid re-optimization of a complex model.

Overview & Example

Example: problem $P$ is one of energy production planning. All days are similar, the network is mostly unchanged, only the energy demands vary. recurrent combinatorial optimization.

- An anomaly occurs, some parameters can change: new problem $P'$, re-optimization is necessary.
- Response time is limited: resolution time budget.
- Variations on a theme: generative process.
- Past resolutions, data available.
- Learn from the past, solve future instances faster.

Methods developed:

1. NaïBRec: block irrelevant variables, only one SP generation.
   - Given $p$ variables, SP: smaller sub-problem, manageable approximation.
2. SuSPen: block/unblock variables, iterative SP generation.
   - Given $p$ variables, SP repeatedly, until resolution time is available.

Blocked variables are assigned a value heuristically. E.g., the optimal solution $x^*_P$ to a reference problem $P_{ref}$, such as the model under nominal parameters.

Mixed Integer Programming

- A mixed integer programming (MIP) Problem $P$ is:
  $$z^* = \min \{c^T x \mid x \in \chi\}$$
  where $\chi = \{x \in \mathbb{R}^p \times 2^\lambda, \text{s.t. } Ax \leq b\}$.
- We define a sub-problem $SP$ of $P$ as:
  $$z_{SP} = \min \{c^T x_{SP} + \epsilon \mid x_{SP} \in \chi\}$$
  const
- block $\chi$ (set of) variables to a reference value: $x_{SP} = \{x \mid x := x_{ref}\}$.
- $x_{SP} = x_{SP}$, $x_{ref}$ are the remaining free variables.

MIP is a widely adopted formulation for combinatorial optimization problems, solved via Branch & Bound algorithms. Highly performant solvers exist (CPLEX, GUROBI), for reasonably sized instances.

- Intrinsic exponential complexity: won’t go away.

(1) NaïBRec: Naïve Bayes for Recurrent Problems

How to block variables? Frame as a multi-label classification problem, find variables not affected by random events.

Multi-label Classification (MLC)

- Image annotation:
  $$\Rightarrow \{\text{beautiful, mountains, Dolomites, go, holiday, summer, winter}\}$$
- Genetic data: classification of gene functions.
  $$\Rightarrow \{\text{cell growth, cell multiplication, structural function}\}$$
- Labels set $\{\text{decision variables}\}$
  $$\Rightarrow \text{How many labels to include?} \Rightarrow \text{How many variables to block?} \Rightarrow \text{Which labels to choose?} \Rightarrow \text{Which variables to block?}$$

Note: when variables are too many, use clusters and hierarchies of variables as proxy labels.

NaïBRec MLC Algorithm – Cascade of Predictors

Step (1) Size estimation, $m$: number of labels to be predicted.
Step (2) Sequential label prediction, given the size of the target vector:
$$\forall m \Rightarrow y_1 \Rightarrow y_2 \Rightarrow \ldots \Rightarrow y_{m-1} \Rightarrow y_m,$$

- Each step is a naive Bayes classifier.

NaiBRec Meta-Algorithm

Step (1) Collect past data, train classification model.
Step (2) Extract features from current $P'$, e.g., differences $P' \Rightarrow P_{ref} \Rightarrow A(MIP – parameters)$.
Step (3) $N_{NaïBRec}(\Delta(\text{parameters})) \Rightarrow \{x_{SP}, x_{SP}\}$, predict variables to be blocked.
Step (4) Generate SP
Step (5) Run optimization

(2) SuSPen, Supervised Sub-Problem Generation

Extends NaïBRec, introducing the concept blocking/unblocking decision variables. It explores the solution space while handling small problems. It respects the time constraint imposed.

Perspectives

- Considering the block/unblock process as a Markov Decision Process (MDP).
- Reinforcement Learning framework.

Acknowledgments

This research benefited from the support of the “FMIH Program Gaspard Monge in optimization and operations research” and from the support to this program from EDF.