Model Evaluation
Variable Domains

• Feature example: \( \pi \)

Identify instances where the approach can be beneficial

• Energetic Reasoning (ER): propagator for the cumulative constraint.

\[ \forall t = 0, \ldots, coh \sum_{x_i \subseteq x < t, a_i} r_{ik} \leq cap_k \]

• High time complexity (\( O(n^3) \)) but more inferences than most other propagators.

→ used only if \( O_{ER}(D_i) = true \) to keep higher inference and reduce time.

• \( \pi_1 \): TimeTabling (\( O(n^2) \)), \( \pi_2 \): ER

• Feature example: average domain tightness (\( O(n) \))

\[ \frac{1}{H} \sum_{i=1}^{n} lat_i - est_i \]

where \( H \) is the current horizon.

Oracle Estimator

\[ O_{\pi}(D_i|x_i \in S(c)) = \begin{cases} 
true & \text{if some value will be pruned} \\
false & \text{otherwise} 
\end{cases} \]

Variable Domains
Feature Computation
Model Evaluation

• Oracle estimated using Machine Learning

• Feature computation and model evaluation must be cheap

Post Fix Point Procedure

• \( cost_{O_\pi} \) < \( \Delta_{cost} \)

• After initial fix point, \( O_\pi \) is consulted until a new fix point is reached

Case Study: Energetic Reasoning

• Energetic Reasoning (ER): propagator for the cumulative constraint.

\[ \forall t = 0, coh \sum_{x_i \subseteq x < t, a_i} r_{ik} \leq cap_k \]

• High time complexity (\( O(n^3) \)) but more inferences than most other propagators.

→ used only if \( O_{ER}(D_i) = true \) to keep higher inference and reduce time.

• \( \pi_1 \): TimeTabling (\( O(n^2) \)), \( \pi_2 \): ER

• Feature example: average domain tightness (\( O(n) \))

\[ \frac{1}{H} \sum_{i=1}^{n} lat_i - est_i \]

where \( H \) is the current horizon.

Good Prediction can be performed

• Training set: random subset of nodes in a search tree where

  -- ER is applied with a probability 0.5

  -- Binary lexicographic branching

• Test set: complete search tree where \( O_{\pi} \) are used

• BL instances (\( A, B \) is cumulative number \( B \) of instance number \( A \))

\[ \begin{array}{c}
\hline
 Training set size \\
\hline
 0.0 \% & 20.0 \% & 40.0 \% & 60.0 \% & 80.0 \% \\
\hline
 2.0 \% & 60.0 \% & 80.0 \% & 90.0 \% & 95.0 \% \\
\hline
 3.0 \% & 60.0 \% & 80.0 \% & 90.0 \% & 95.0 \% \\
\hline
 4.0 \% & 60.0 \% & 80.0 \% & 90.0 \% & 95.0 \% \\
\hline
 5.0 \% & 60.0 \% & 80.0 \% & 90.0 \% & 95.0 \% \\
\hline
\end{array} \]

Future work

• Prediction performances must take the "benefit in time" of a node into account

  -- E.g., depth of a node.

• If a subtree can be explored faster with \( \pi_1 \) than with \( \pi_2 \) but still \( O(\pi_2) = true \), we should use \( \pi_1 \) (other kind of prediction).

References

