APPRIL Integration Status

July 12\textsuperscript{th}, 2015

Junkyu
APPRIL Integration Status

**DAOOPT for MPE**
- CVO → variable ordering
- GLS+ → Lower Bound
- FGLP/JGLP → Upper Bound
- MBE-MM → heuristic
- BRAOBB → Search

**Other Solvers**
- Bucket Elimination
- MEX: Belief Propagation
- stand alone GLS+

**Integration of GM Algorithms (DAOOPT/BE/MEX/GLS+)**
- MPE Task: DAOOPT/GLS+ / BE (bucket elimination)
- PR Task: BE / MEX BP (belief propagation)

**Supporting Windows/ LINUX environment**
Prediction of Problem Hardness (1/3)

**Analysis (preprocessing)**

1. Find Variable Ordering
2. **Select Parameters**
3. GLS+ : Find Lower Bound
4. FGLP/JGLP : Cost shifting
5. MBE-MM : Static Heuristic
6. **Estimate Search Space**

**Query (search)**

Find MPE solution

• **Selection of Algorithm Parameters**
  – Find a set of algorithm parameters for preprocessing components
    • Guided Local Search + : Max. iteration / time
    • FGLP, JGLP : Max. iteration/ time
    • MiniBucket Elimination + Moment Matching : I bound
  – SVM classifier
    • Predict a class label indicating the hardness
    • Use predetermined algorithm parameters for each class

• **Estimation of the Search Space**
  – Estimate the number of nodes in search space by Linear Regression
Prediction of Problem Hardness (2/3)

- **Learning SVM Classifier**
  - **Data:** PASCAL2 Benchmark (7 domains/930 instances)
  - **Label instances by the best time performance:** ($<$ 5sec, $<$ 20sec, $<$ 20min, $<$ 2hr)
  - **27 static (problem intrinsic) features**
    - 9 static features by running schematic Bucket Elimination
    - 6 static problem features (N,F,K,S,W,H)
    - 12 derive features ($W*\log(K)$, $H*\log(K)$, $W*\log(S)$, $H*\log(S)$, etc)
  - **Run **libsvm 3.20**
    - [https://www.csie.ntu.edu.tw/~cjlin/libsvm/](https://www.csie.ntu.edu.tw/~cjlin/libsvm/)
    - Use default options, Gaussian Kernel, 5 fold cross validation, grid search for parameters

<table>
<thead>
<tr>
<th>Labels</th>
<th>i bound</th>
<th>gls+ iter</th>
<th>gls+ time</th>
<th>FGLP iter</th>
<th>FGLP time</th>
<th>JGLP iter</th>
<th>JGLP time</th>
</tr>
</thead>
<tbody>
<tr>
<td>trivial (&lt;5 sec)</td>
<td>max - 3</td>
<td>1</td>
<td>1 sec</td>
<td>-</td>
<td>1 sec</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>easy (&lt;20 sec)</td>
<td>max - 1</td>
<td>1</td>
<td>2 sec</td>
<td>-</td>
<td>5 sec</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>moderate (&lt;20 min)</td>
<td>max</td>
<td>10</td>
<td>10 sec</td>
<td>1000</td>
<td>30 sec</td>
<td>500</td>
<td>30 sec</td>
</tr>
<tr>
<td>hard (&lt;2 hr)</td>
<td>max</td>
<td>20</td>
<td>20 sec</td>
<td>2000</td>
<td>60 sec</td>
<td>1000</td>
<td>60 sec</td>
</tr>
<tr>
<td>2 hr time out (time out)</td>
<td>max</td>
<td>20</td>
<td>40 sec</td>
<td>4000</td>
<td>120 sec</td>
<td>2000</td>
<td>120 sec</td>
</tr>
<tr>
<td>undefined</td>
<td>max -1</td>
<td>1</td>
<td>2 sec</td>
<td>-</td>
<td>5 sec</td>
<td>-</td>
<td>-</td>
</tr>
</tbody>
</table>

- jglp: terminate when reaching max. iteration or time
- sls/mplp: allow early termination by convergence before reaching max. iteration or time
Prediction of Problem Hardness (3/3)

Learning Linear Regressor

- 26 static features
  - number of variables, pseudo tree depth, number of leafs in pseudo tree, etc
  - min/max/average/std.dev of static features
  - i-bound, maximum context size – i bound

- 9 dynamic features
  - bounds : from the results of GLS+/FGLP/JGLP/MBE-MM
    - lower bound, upper bound, UB-LB
  - sample search space: probe search space (20000 nodes default)
    - Ratio of purning (by heuristic / determinism)
    - Average depth/branching from probe

- Regression Learning by Lasso
  (LassoLars(alpha=0.01), degree=1, (10603 samples)
  \[
  \log(\text{num\_nodes\_in\_Search\ Space}) = \\
  (1.64620\text{e-04} * (ub)) + (3.83243\text{e-01} * (ub-lb)) + \\
  (6.77123\text{e-02} * (avgNode\_Depth)) - (4.05910\text{e-02} * (Depth)) + \\
  (3.78208\text{e-03} * (Vars)) - (7.44384\text{e-03} * (Leafs)) + \\
  (4.6389\text{e-01} * (W\_avg)) + (2.47618\text{e-01} * (W\_std\_dev)) - \\
  (1.68938\text{e-01} * (Context\_max)) + (9.91102\text{e-02} * (H\_avg)) - \\
  (2.57769\text{e-01} * (H\_std\_dev));
  \]

Learning a Regression Model for Complexity Estimation (Otten and Dechter, 2012)

- Number of nodes $N(n)$ as linear function of features $\varphi_j(n)$:

$$\log N(n) = \sum_j \lambda_j \varphi_j(n)$$

related: [Leyton-Brown, Nudelman, Shoham 2009]
Subproblem Features $\varphi_j(n)$

- Use both static and dynamic characteristics:
  - Structural features:
    - Problem solution cost, derived from the problem instance.
    - Problem solution cost, provided when upper and lower bound exists of the subproblem.
  - Dynamic features:
    - Running time of the heuristic.
    - Use of determinism (zero probability) in the heuristic.
    - Running on running 5000 node expansion point.

Probe of AOBB:
- Average depth of terminal search nodes within probe.
- Average node depth within probe (denoted $d$).
- Average branching degree, defined as $\sqrt{\sqrt{5000}}$.

Static:
- $\lambda$: Mini bucket $i$-bound parameter.
- Max. subproblem variable context size minus mini bucket $i$-bound.
Predicting Depth-First Branch and Bound Search Trees (Levi, Lars and Dechter, IJCAI 2013, CP-2014 submission)