Pearl Hunter: A Cross-domain Hyper-heuristic that Compiles Iterated Local Search Algorithms

7th September 2011, Nottingham, UK

Fan Xue, CY Chan, WH Ip, CF Cheung
Department of Industrial & Systems Engineering
Hong Kong Polytechnic University
Outline

1. Introduction
2. Pearl Hunter
3. Training and Validation on HyFlex
4. Migrating to Quadratic Assignment
5. Discussion and conclusion
Ancient stories (intelligence & optimization)

- Carthaginian bull’s hide
  - (In 810s B.C.,) … Elissa (Dido) had an bull’s hide cut into strips and lay them out end-to-end in a *crescent* circumscribing a sizeable area of land. This ox-hide enclosed area was known as Carthage.

- Dido’s problem
  - \[
  \begin{align*}
  \text{max} & \int_a^b f(x) \, dx \\
  \text{s.t.} & \int_a^b \sqrt{1 + (f'(x))^2} \, dx = \text{Len}_{\text{oxhide}} \\
  f(a) &= f(b) = 0
  \end{align*}
  \]
  - The correct answer is Dido’s *semicircle*, and \(d(a, b)\) is the diameter.
Ancient stories (intelligence & optimization)

**Tian Ji’s racing horses**

“(In 340s B.C.,) General Ji Tian of Kingdom Qi raced horses with other members of royal family several times. His guest Bin Sun (author of Sun Bin’s Art of War) found Tian’s 3 horses covered 3 levels and were not much inferior in races…” (Sima, 91bc)

**Sun’s strategy**

- Displacement: sent Tian’s inferior horse (T_c) to race in the name of the best (T_a), T_a to race in the name of the average(T_b), T_b to race in the name of the inferior (T_c).

- Tian and King Wei of Qi had a horseracing. Tian won 2/3 races, and won a prize of about 500 oz copper. Sun became the military counselor of Qi.
Hyper-heuristics

- Enormous of optimization methods have been proposed so far. Four issues have been concerned:
  - Effectiveness able to find highly satisfactory solutions,
  - Efficiency with quick running,
  - Easiness (IMO) easy to understand and deploy, and
  - Portability scalable to different domains and datasets.

- Machine Learning: main source of the portability power

- Hyper-heuristics \( \subseteq (?) \) Metaheuristics
  - Hyper-heuristics select or generate heuristics via online or offline learning, to combine the strength and to compensate the weakness of each “low-level” heuristic (like Sun’s?), if each heuristic has its own strength and weakness.

```
   D  A  D
  Selection  online /offline

   A  B  C  D  E  F

  Generation  online /offline

   X  Y  X
Low-level heuristics
```
Pearl Hunter: an inspiration

- **Pearl hunting** is an out-of-date diving activity of retrieving pearls from oysters. Can still be found in:
  - Some Asia tourist sites,
  - Virtual games.

- In a search perspective, pearl hunting consists of repeated
  - *diversification* (surface and change target area)
  - *intensification* (dive and find pearl oysters).

- Pearl hunting is in the paradigm of Iterated Local Search (Lourenço *et al*, 2003).

Pearl hunting in World of Warcraft
Pearl Hunter: a hyper-heuristic imitation

- Basic actions made of low-level heuristics
  - **Snorkeling**: local search with a low “depth of search”, stops after any improvements
  - **Deep dive (SCUBA)**: local search with a high “depth of search”, till no further improvements
    - Why two intensifications? $N_{\text{snorkeling}}/N_{\text{dive}}$?
  - **Surface moves**: non-local-search heuristics
    - **Crossover** (MI): 2 or more input solutions
    - **Mutation** (SI): 1 input solution

- “Environment”:
  - **Shallow water**, where local search is useless
  - **Sea trench**, where local search costs too much time
  - **Default**, otherwise

---

XUE et al: Pearl Hunter: A Cross-domain Hyper-heuristic
Pearl Hunter: a hyper-heuristic imitation (Continued)

- Pearl Hunter can drop a *Buoy* at the depth of first deep dive, to escape from local optimum by mutations (SIs).

- Four running modes (portfolios) of moves:
  - **A**: all moves averagely, with a *Buoy* mark
  - **B**: *crossover* with a *Buoy* mark (triggering a few mutations)
  - **C**: *crossover* only, no mutation, no *Buoy*
  - **D**: Sea trench mode, all surface moves averagely, no *Buoy*.

Moves are subject to online pruning.

- Other tricks:
  - **tabu lists** (memory), “mission restarts” (go to new areas)
1) Selecting low-level heuristics and 2) determining one mode after the “perceiving” period (classification)

- Off-line learning
  - Rule induction
  - …

- Online learning
  - Full problem
  - A division of problem

Attributes for offline learning:
- Suboptimal solutions found by moves, dives
- Restarts, MI vs SI, …

A “cropped” problem with a subset of given variables, smaller but “keeps flavor” (good in division, good in problem).

Example: a division of a TSP (u159)
HyFlex and CHeSC

- HyFlex (Hyper-heuristics Flexible framework) is a java cross-domain platform (Burke et al, 2011)
  - 6 domains, 4 public (training domain) and 2 hidden
  - “Black-box” low-level heuristics in 4 categories:
    - Crossover, Mutation, Ruin-recreate, and Local search
  - Parameters to control low-level heuristics:
    - “Intensity" of mutations, and “depth of local search”

- CHeSC 2011 is the first Cross-domain Heuristic Search Challenge on HyFlex. (http://www.asap.cs.nott.ac.uk/chesc2011/)

- Pearl Hunter was ranked in CHeSC:
  - 4th out of 20 entries overall,
  - 1st out of 20 entries in the hidden domains.
HyFlex and CHeSC: BF-Tree obtained by offline learning (by Weka v3.5)

- $D_{murr}$: Depth of the mission in the Mutation and Ruin-recreate test,
- $M_{co}$: Number of missions completed in the Crossover test,
- $N$: Number of sub-optimal solutions found in total,
- $P_{dir}$: Percent of sub-optimal solutions found right after some moves (before any dive),
- $P_{mu}$: Percent of sub-optimal solutions found in iterations started with Mutation moves,
- $P_{rr}$: Percent of sub-optimal solutions found in iterations started with Ruin-recreate moves,
Tests on personnel scheduling: beyond the 600s time limit of CHeSC

- On large-scale personnel scheduling problems,
  - Running time was increased to 10 hours (normalized to P4 3GHz),
  - Same decision tree

- New best known solutions:

<table>
<thead>
<tr>
<th>Instance</th>
<th>Men</th>
<th>days</th>
<th>Time (h)</th>
<th>Result</th>
<th>Prev BK*</th>
<th>% improved</th>
</tr>
</thead>
<tbody>
<tr>
<td>CHILD-2A</td>
<td>41</td>
<td>42</td>
<td>10</td>
<td>1,095</td>
<td>1,111</td>
<td>1.4</td>
</tr>
<tr>
<td>ERRVH-A</td>
<td>51</td>
<td>42</td>
<td>10</td>
<td>2,142</td>
<td>2,197</td>
<td>2.5</td>
</tr>
<tr>
<td>ERRVH-B</td>
<td>51</td>
<td>42</td>
<td>10</td>
<td>3,121</td>
<td>6,859</td>
<td>54.5</td>
</tr>
<tr>
<td>MER-A</td>
<td>54</td>
<td>42</td>
<td>10</td>
<td>9,017</td>
<td>9,915</td>
<td>9.1</td>
</tr>
</tbody>
</table>

* Best known values were collected from [http://www.cs.nott.ac.uk/~tec/NRP/misc/NRP_Results.xls](http://www.cs.nott.ac.uk/~tec/NRP/misc/NRP_Results.xls)

- A possible reason
  - A new “vertical” swap concept first implemented in low-level heuristics on HyFlex
QAP: Another test domain

- Quadratic assignment problem (QAP)
  \[ \sum_{a,b \in F} \omega_{a,b} \cdot d_{A(a), A(b)} \] where A is assignment.
  - Example: place N facilities in a grid of cellular manufacturing (facility layout problem).
  - NP-hard

- Coded as a new domain on HyFlex

- Low-level heuristics implemented
  - Crossover heuristics
    - Partially Matched Crossover (Chan and Tansri, 1994)
    - Order Crossover (Chan and Tansri, 1994)
    - A voting recombination crossover
QAP: Low-level heuristics (continued)

- Mutation heuristics
  - Random swaps
  - Shifting mutation (PSSC Lab, 2005)
  - Spiral reassignment (Yaman et al, 1993)

- Ruin-recreate heuristics
  - Chan’s heuristic (Chan et al, 2002)
  - GRASP (greedy randomized adaptive search procedure, Feo and Resende, 1995)

- Local search heuristics
  - Variable Depth Search with partial gains (Burke et al, 2007)
  - Tabu Search (Taillard, 1991)

- Division selection heuristic
  - Selecting flow and distance matrices with closest means and deviations from 1000 random division samples.
QAP: Experiments

- **Instances**
  - 10 largest from QAPLIB, Euclidean and non-Euclidean

- **Algorithms**
  - **PHunter**: Codes and rules for CHeSC (no modification)
  - **PHunter$_5$**: Same codes, Appended QAP to training domains (4->5)
  - **PHunter$_{div}$**: Same codes, CHeSC rules, appended a simple online mode learning via a division ($N' = 0.47*N$) : try different modes independently (5% time, pow(5%, 0.25)=0.47), chose the best one.
  - **NIFLS_LS**: Iterated local search (Ramkumar *et al*, 2009).
  - **ILS**: Iterated Local Search (Stützle, 2006)
  - **PMA-SLS$_{10}$**: Parallel Memetic Algorithm with Selective Local Search (10 islands, Tang *et al*, 2006)
QAP: Experiments (continued)

Off-the-peg Hunters versus custom-made methods on $N=100$ instances (10 independent runs, 600s for each run, time normalized to a P4 3.0GHz)
# QAP: A close look of results

Average objective values excess best-known values (\%)
Discussion

- The off-the-peg hyper-heuristics can be comparable to the custom designed metaheuristics

- Results of PH, PH₅, PH₅ᵥ are very close
  - PH: Pearl Hunters seem portable without tweak on codes
  - PH₅ᵥ: Capability of learning online from a proper division

- Difficulties in getting size of division N’:
  - N’ could be determined by: \( \frac{t_{LS}(N’)}{t_{LS}(N)} = \frac{t_{perceiving}}{t_{hunting}} \)
    - Proportions of local search may change, e.g., \( t_{LS1} = O(N^5) \), \( t_{LS2} = O(N^2) \)
  - Complexity of division is still not well redressed by the equation. (NP-hardness versus polynomial algorithms)
    - N’ too small, easy to find optimum, unable to rank heuristics;
    - \( \frac{t_{perceiving}}{t_{hunting}} \) larger, less time for hunting.
Eventually a “point and shoot” hyper-heuristic software for daily use?

- Step 1: Define variables, an objective function, and constraints,
- Step 2: Solutions out.

Cross-domain crossover, mutation, local search?

More accessible function on HyFlex?

- Such as “Similarity between two solutions”

“Learn-and-generate” hyper-heuristic on HyFlex?

- Encapsulate training data \(<\text{attribute1}_i, \text{value1}_i, \text{attribute2}_i, \text{value2}_i, \ldots, \text{label}>\) for each low-level heuristic \(i\).
Conclusion

- We present a hyper-heuristic
  - Imitates pearl hunting
  - Perceives “environment” of search
  - Determines a perturbation mode by online/offline learning
  - Generates different modes of ILS
- We find the results of tests encouraging
- Possible future works
  - Other reasonable ways to classify mode online
  - Hunters can generate new low-level heuristics
    - (Custom designed for TSP) Generated an association-rules-based weighting heuristic to determine candidate set, and facilitated branch-and-bound and local search (LKH) (Xue et al, 2010).
References


QAPLIB: http://www.seas.upenn.edu/qaplib/


Thank you for your attention!

E-mail addr.: mffxue@inet.polyu.edu.hk
dewolf_matri_x@msn.com