META-HEURISTICS DEVELOPMENT FRAMEWORK: DESIGN AND APPLICATIONS

MASTER THESIS PRESENTATION
28th of July, 2004

WAN Wee Chong, Jason
National University of Singapore
School of Computing (SoC)
AGENDA

- **Introduction**
- **Literature Reviews**
- **Meta-heuristics Development Framework**
  - Components
  - Search Strategies
- **Case Study:** The Traveling Salesman Problem
  - Hybridized Schemes
  - Observations of Results
- **Project Summary**
- **Questions**
INTRODUCTION
INTRODUCTION

Meta-heuristics

- **Traditional Methods**
  - Inadequate at solving large-scaled combinatorial optimization problems
  - Meta-heuristics matured rapidly as a solution

- **Popular Meta-heuristics**
  - E.g. Tabu Search, Simulated Annealing, Genetic Algorithms, Constraint Local Search

- **New Meta-heuristics**
  - E.g. Ants Colony Optimization, Path Re-linking

- **Potential of hybridization**
  - Diverse growth of meta-heuristics of various nature
  - Utilize the forte of meta-heuristics
Introduction

- Conventional Development Approach
  - Developing from Scratch
  - Waste of resources (Man and Machines)
  - Absence of a standard template for benchmarking

- Demand for an efficient development tool
  - Major reduction in developing time
  - Standard platform for implementation and benchmarking
  - Object oriented (discipline, ease of integration and extension)
  - Facilitate rapid prototyping of new techniques
Design Goals

- **Generic**
  - Work with most if not all combinatorial optimization problems
  - Able to model various search strategies
    - E.g. Hybridization, Intensification, Diversification

- **Reusability**
  - Offload repetitive search routines
  - Reuse both design and codes

- **Clarity**
  - Unambiguous interfaces that gives clarity
  - Allow rapid implementation of application
**Comparison factors**

1. Usage Friendliness
2. Number of meta-heuristics supported
3. Clarity in Design
4. Adaptive Control
5. Ease of Hybridization
6. Extensible Library

---

**MDF**
- Lau and Wan, 2004

**Open TS**
- Robert Harder, 2003

**Localizer ++**
- Michel and Hentenryck, 2001

**Easy Local ++**
- Gaspero and Schaef, 2001

**Hot Frame**
- Fink and Voß, 2002
MDF Features

- C++ Meta-heuristics framework
- Adopt object-oriented design (OOD)
  - Using interfaces to achieve genericity
- Currently supported four (4) meta-heuristics
  - Tabu Search, Ants Colony Optimization, Simulated Annealing and Genetic Algorithm
- Centralized control mechanism for adaptive search
  - Using Request and Response Metaphor
  - Model into an “event-driven” search
- Include a software library
  - Speed-up development progress
  - Reduce coding errors
**Open TS Features**

- Initialized by **Computational Infrastructure for Operation Research (COIN-OR)**
- Java-based Tabu Search
- Generic aspect achieved through interfaces
- Unambiguous interfaces that define clearly their collaborative roles in the algorithm
  - Solution, Move Manager, Move, Objective Function, Tabu list
- Support adaptive control through decentralized *Event Listeners*
- Limited support of software components
  - Simple/Complex Tabu List
  - Complex Move

**Usage**
- **Adaptive Control**
- **Ease of Hybridization**
- **Clarity in Design**
- **Meta-heuristics supported**
- **Extensible Library**
- **Friendliness**
- **Ease of Hybridization**
- **Clarity in Design**
- **Meta-heuristics supported**
- **Extensible Library**
Localizer ++ Features

- C ++ Constraint-based Local search framework
- Require formulation of problem into its mathematical equivalent
  - E.g. Decision Variables, Objective Functions and Constraint Equations
- Has a two-level architecture
  - *Declarative components*: for data storage management
  - *Search components*: for defining search procedure
- Search components can incorporate various local search algorithms
  - Neighborhood operators
  - Tabu Lists
- Extensible constraint library
  - All-diff constraints

Usage

- Friendliness
- Adaptive Control
- Ease of Hybridization
- Clarity in Design
- Meta-heuristics supported
- Extensible Library
**Easy Local ++ Features**

- C ++ Local search framework
- Adopt a four level architecture to implement local search techniques
  - *Basic Data*: for maintaining the states of search space
  - *Helpers*: for handling search actions such as exploration of neighborhood
  - *Runners*: for performing the routine of the meta-heuristics using the helpers
  - *Solvers*: for generating the initial solutions
- Additional support classes
  - *Kickers*: for diversification
  - *Testers*: for debugging
- Support limited hybridization
  - E.g. SA as diversifier
- Absence of component library

---

**Usage**

- **Friendliness**
- **Adaptive Control**
- **Ease of Hybridization**
- **Clarity in Design**
- **Extensible Library**
- **Meta-heuristics supported**
- **Usage**
  - **Hybridization**
  - **Clarity in Design**
  - **Extensible Library**
  - **Meta-heuristics supported**
  - **Ease of Hybridization**
  - **Adaptive Control**
  - **Friendliness**
Hot Frame Features

- C++ Meta-heuristics framework
- Support various meta-heuristics and their derivations
  - Tabu Search, Simulated Annealing, Evolutionary Algorithms
- Use template design for meta-heuristics routines
- Use inheritance to override the meta-heuristics procedures
- Provide general software components
  - Reusable data structure classes
    - E.g. binary vectors, permutations, combined assignment and sequencing
  - Standard neighborhood operators
    - E.g. bit-flip, shift, swap moves
- Inflexibility in codes recycling

Usage
- Friendliness
- Adaptive Control
- Ease of Hybridization
- Clarity in Design
- Extensible Library
- Meta-heuristics supported
DESIGN AND ARCHITECTURE
**General routines of Meta-heuristics**

1. **New State [Solution]**
2. **Generate Next State [Neighborhood Generator]**
3. **Check Feasibility [Constraint]**
4. **Evaluate State [Objective Function]**
5. **Apply Penalty [Penalty Function]**
6. **Translate to New State [Move]**

**Control Mechanism**

**Execution Flow [Engine]**

**Unique Behavior [Proprietary Actions]**
Overview of MDF

- Control Mechanism
  - Event Controller
  - Lists of Handlers
  - Lists of Events

- Resource Container
  - General Controls
  - TS Controls
  - ACO Controls
  - SA Controls
  - GA Controls

- Engine Components
  - TS Engine
  - ACO Engine
  - SA Engine
  - GA Engine

- Proprietary Interfaces
  - Tabu Search
  - Ants Colony
  - Simulated Annealing
  - Genetic Algorithm

- General Interfaces

<table>
<thead>
<tr>
<th>Solution</th>
<th>Neighborhood Generator</th>
<th>Objective Function</th>
<th>Penalty Function</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Move</td>
<td>Constraints</td>
<td></td>
</tr>
</tbody>
</table>
COMPONENTS OF MDF

- **General Interfaces**
  - Solution, Move, Constraint, Neighborhood Generator, Objective Function, Penalty Function

- **Proprietary Interfaces**
  - Tabu List and Aspiration Criteria
  - Pheromone Trails and Local Heuristics
  - Annealing Schedule
  - Recombination and Population

- **Engines**

- **Control Mechanism**
  - Event
  - Handler
  - Event Controller
Solution Interface

- Solution Representation
  - Make no assumption on the data structure type
  - Data are manipulated indirectly through other interfaces

2D Arrays

Arrays of Lists

Array of Boolean

Cloning Function

- Shallow cloning: Copying the reference of the object
- Deep cloning: Copying the contents of the object
Move Interface

- Define the neighborhood
  - **TS:** Local Improvement
    - E.g. Exchange, Replace
  - **ACO:** Constructing solution
    - E.g. Increment
  - **SA:** Probabilistic operation
    - E.g. Probabilistic Swap
  - **GA:** Recombination
    - E.g. One-point crossover
Constraint Interface

- Measure the degree of violation
  - Boolean: Feasible / Infeasible
  - Integer: 0 = Feasible, 1 – n = number of constraints violated

- Useful for
  - Oscillating strategies
  - Dual Model formulation
  - Ranked (hierarchical) constraints based search
Neighborhood Generator Interface

- Generate the possible next states
- Use Move and Constraint
  - Move: Generates all possible moves
  - Constraint: Ensure the moves are desirable

- Has a different “contextual meaning” for different meta-heuristics
  - TS: Generate all neighboring solutions
  - ACO: Generate the next possible paths
  - SA: Generate a random neighboring solution
  - GA: Generate the probability table for next generation selection
Objective Function Interface

- Compute the objective value of solution
  - **Absolute** Computation
    - Calculate objective value from scratch
  - **Incremental** Computation
    - Calculate objective value from the previous solution
    - Usually applied to calculate the neighbor objective value from current solution

- Compare the objective values of two solutions
  - Determine **maximizing** or **minimizing** the objective value
**Penalty Function** Interface

- Implementation of *soft constraints*
  - Some solutions are preferred even if their objective value is slightly lower

- Objective value
  - Temporary modified during comparison
  - Prevent re-application of penalty (bonus)
  - Maintain the correctness of the objective value
Tabu List Interface

- Memory technique generally used to reduce cycling
  - Short tenure leads to cycles
  - Long tenure decreases efficiency

Moving downhill

Greedy downhill approach
Aspiration Criteria Interface

- Override tabu status of a neighbor if it meets certain criteria
  - Objective value is better than best-found solution
  - Difference between current and next states is large
  - Existence of certain sub-optimal structures

- Avoid accidentally missing good solutions
- Improve the search performance
Pheromone Trails Interface

- Record the pheromone density of the ants trails
- Influence the behavior of subsequent ants

Pheromone Update

- **Local** Decay
  - Enhance exploration
- **Global** Update
  - Enhance exploitation
- **Evaporation**
  - Reduce rapid convergence

Number of Ants

Iteration To Go
Local Heuristics Interface

- Incorporate the underlying heuristic in solving problem
- Reflect the quality of the path
- E.g. in TSP, local heuristic compute the quality of an arc
  - \( Q \text{(arc)} = \frac{1}{\text{distance}} \)
- Can be formulated as a function of multiple factors
  - E.g. Vehicle Routing Problem with Time Windows
    - Vehicle
    - Distance
    - Waiting Time
Annealing Schedule Interface

- Acceptance Probability $p$
  
  $p = \text{exponential} (-|\Delta x/T_i|)$

- $\Delta x$: difference in objective values of current and next states

- $T_i$: cooling temperature

Modeling $T_i$

- If $T_i = 0$, the algorithm becomes greedy
- If $T_i = \infty$, the algorithm becomes random
- $T_i$ is usually set to $\infty$ and gradually decrease to 0
Recombination Interface

- Combine two individuals to produce two offspring
- Crossover operators
  - One-point crossover
  - Two-point crossover
  - Uniform crossover
  - Partially mixed crossover
- May incorporate a probability of crossover
  - Encode the probability of two individuals will actually breed
Population Interface

- Contain individuals in a generation
- Generate the First Generation pool
  - Created randomly or by randomized heuristics
  - Ensure diversity so as to prevent rapid convergence
- Selection of subsequent generation
  - Sometimes parents are also preserved in the next generation
    - Prevent the loss of “good” genes
    - Lead to over-population
  - Discard the unwanted individuals in the mixed pool
    - “Survival of the fittest” rule
    - Rules can be specified by users
Engines

Tabu Search Engine

procedure
Initialize a current Solution
while terminating criteria not reached

- Neighborhood Generator generates a new neighborhood;
- Constraint discards any undesired neighbors;
- Objective Function evaluates selected neighbors;
- Penalty Function applied to neighbors;
- Tabu List and Aspiration Criteria are consulted;
- Move translates current Solution to best neighbor;
- if new Solution is better than best found Solution
  Clones and records new Solution as best found Solution;
- Tabu List is updated;
end procedure
Ant Colony Optimization Engine

procedure
Initialize the Pheromone Trail
while terminating conditions not reached
  while there is still ants in colony and
    while the Solution is not completed
      Neighborhood Generator generates a set of new trails;
      Constraint discards any impassible trails;
      Trail chosen by consulting Local Heuristic and Pheromone Trail
      Move translates the Solution with selected trail;
      Local Pheromone Trail Updated
      Objective Function evaluates solutions constructed by ants;
      Penalty Function is applied to determine the quality of solutions;
      Global Pheromone Trail is updated;
      If new Solution is better than best found Solution,
        Clones and records new Solution as best found Solution;
      Evaporation occurred in Pheromone Trail;
end procedure
Simulated Annealing Engine

procedure
Initialize a current Solution;
while terminating conditions not reached
    Neighborhood Generator generates a random neighbor;
    Constraint validates the feasibility of neighbor;
    Objective Function evaluates solutions;
    Penalty Function temporary adjusts the objective value;
    If new neighbor is better than current Solution
        Move translates Solution to neighbor;
    Else
        Consults the Annealing Schedule;
            If neighbor is accepted
                Move translates Solution to neighbor;
            Else
                Current Solution remains unchanged;
    If new Solution is better than best found Solution
        Clones and records new Solution as best found Solution;
end procedure
Genetic Algorithm Engine

procedure
Initialize the first generation Population;
while terminating conditions not reached
  Neighborhood Generator selects Solutions for mating;
  Recombination crosses selected Solutions to form new children;
  Move mutates new children;
  Constraint discards infeasible children;
  Objective Function evaluates children;
  Children are mixed into the parent Population;
  Penalty Function adjusts the objective value of all Solutions in Population;
  Population discards unfit individuals until the population is balanced;
  If any Solution in Population is better than best found Solution
    Clones and records new Solution as best found Solution;
end procedure
Inspire from observing that search strategies enhance meta-heuristic performance

- Intensification, diversification, hybridization
- Provide strong motivation for strategy-based framework

**ALL** strategies are defined by two components
1. The time point in which the strategy is to be performed
2. The actions performed by the strategy

- **Request** and **Response (R&R)** Metaphor
  - Requests are specific time points in the search (Event)
  - Responses are actions to be performed (Handler)
  - The search process becomes an “event-driven” simulation
**Atomic unit time concept**

- The smallest unit time for a meta-heuristic to completely perform a set of routines
- Definition varies across different meta-heuristics

<table>
<thead>
<tr>
<th>Meta-heuristics</th>
<th>Atomic unit time Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tabu Search</td>
<td>An iteration of the search</td>
</tr>
<tr>
<td>Ants Colony Optimization</td>
<td>The activity of an ant</td>
</tr>
<tr>
<td>Simulated Annealing</td>
<td>Generating a random move</td>
</tr>
<tr>
<td>Genetic Algorithm</td>
<td>A new generation formed</td>
</tr>
</tbody>
</table>
Event Interface

- Implement the requests of user
  - New Best Solution Found
  - Series of Non-improving Moves

- Many-to-many relationship between events and handlers
  - One event can trigger multiple responses
  - Many events can trigger a same response

- Support three levels of priority
  - **INSTANT:** Execute the handler immediately
  - **NORMAL:** Execute the handler at the end of atomic unit time
  - **DELAYED:** Execute after all higher priority handlers are executed

- Hierarchical levels of priority allows user to control precisely the sequence of execution
**Handler Interface**
- Implement the responses of user
- **Parameters based** handlers
  - Implement adaptive parameters
    - Reactive Tabu List
    - Dynamic Annealing Schedule
- **Techniques based** handlers
  - Incorporate additional objects to handle the actions
    - Intensification on Elite Solutions
    - Probabilistic Diversification
    - Hybridization
Event Controller Class

- Search State
  - Operating meta-heuristic engine
  - Search parameters
  - Current Solution

- Control the search process by adjusting the search state

- Special Case: Hybridization
  - Implement a “Chain of Responsibility”
CASE STUDY: Traveling Salesman Problems (TSP)
Problem Definition

Traveling Salesman Problem (TSP)

Let $G = (V,A)$ be a graph,

where

$V = \{v_1, v_2, \ldots, v_n\}$ be a set of cities (vertex set),

And

$A = \{(v_i, v_j) : v_i, v_j \in V, i \neq j\}$ be the edge set,

Cost $(r, s) = \text{Cost} (s, r) \quad (\text{Symmetry})$

- A tour is defined as a Hamiltonian circuit passing exactly once through each point in vertices $V$.
- The TSP objective is to find a tour of minimum costs/distance
Experiment Platform

- **System Specification**
  - Processor: Athlon XP 3.2 Ghz
  - Memory: 512 MB
  - Runtime: 90 seconds
    - Hybridize Schemes
    - Problem size

- **Operating System**
  - Window XP

- **Initial Solution**
  - Nearest Cities heuristic (Greedy heuristic)
  - Non-optimality of the last portion of the tour
Hybrid Ants System and Tabu Search

- Flexible hybrid scheme that spawns **four derived models**
  - Relative importance level
  - Degree of collaboration
  - “Master-Slave” relationship

<table>
<thead>
<tr>
<th>HASTS-EA</th>
<th>HASTS-IE</th>
<th>HASTS-ED</th>
<th>HASTS-CC</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ants System (Dominant) ➔ Tabu Search (Dominant)</td>
<td>Ants System (Dominant) ➔ Tabu Search (Passive)</td>
<td>Tabu Search (Dominant) ➔ Ants System (Passive)</td>
<td>Ants System (Passive) ➔ Tabu Search (Passive)</td>
</tr>
<tr>
<td>Integration of Tabu List and Pheromone Table</td>
<td>Tabu Search (Passive)</td>
<td>Collaboration via Common objectives</td>
<td></td>
</tr>
</tbody>
</table>

- 11 Model
- 10 Model
- 01 Model
- 00 Model
HASTS Schemes

HASTS Derived Schemes

- Pure TS (Strict Tabu Search)
  - Static tabu tenure
- Pure ACO (Ants Colony System)
  - Update using iteration-best ants
- HASTS-EA (Empowered Ants)
  - Use both pheromone trails and tabu list
- HASTS-IE (Improved Exploitation)
  - Use TS to improve on the solution found by iteration-best ant
- HASTS-ED (Enhanced Diversification)
  - Use ACS as a diversifier
- HASTS-CC (Collaborative Coalition)
  - Two phase search
- Hyper Hybrid Schemes (Combined)
  - HASTS-IEEA (IE with EA)
  - HASTS-CCED (CC with ED)
OBSERVATIONS OF RESULTS

Development Time Comparison

- Development Time Comparison Diagram

- Development Time: 2000, 2200, 2400, 2600, 2800, 3000, 3200

- Categories: MDF, Misc, TS, ACO, Additional Efforts

- Comparison between different categories over time.
## Observations of Results

<table>
<thead>
<tr>
<th>Name</th>
<th>Bound</th>
<th>Pure TS</th>
<th>Pure ACO</th>
<th>HASTS-EA</th>
<th>HASTS-IE</th>
</tr>
</thead>
<tbody>
<tr>
<td>Att48</td>
<td>10628</td>
<td>10755</td>
<td>1.19</td>
<td>10847</td>
<td>2.06</td>
</tr>
<tr>
<td>eil51</td>
<td>426</td>
<td>427</td>
<td>0.23</td>
<td>430</td>
<td>0.94</td>
</tr>
<tr>
<td>Pr76</td>
<td>108159</td>
<td>109186</td>
<td>0.95</td>
<td>111994</td>
<td>3.55</td>
</tr>
<tr>
<td>kroA100</td>
<td>21282</td>
<td>21296</td>
<td>0.07</td>
<td>21559</td>
<td>1.30</td>
</tr>
<tr>
<td>kroB100</td>
<td>22141</td>
<td>22235</td>
<td>0.42</td>
<td>23145</td>
<td>4.53</td>
</tr>
<tr>
<td>Will01</td>
<td>629</td>
<td>629</td>
<td>0.00</td>
<td>649</td>
<td>3.18</td>
</tr>
<tr>
<td>Chl30</td>
<td>6110</td>
<td>6196</td>
<td>1.41</td>
<td>6492</td>
<td>6.25</td>
</tr>
<tr>
<td>kroA150</td>
<td>26524</td>
<td>27125</td>
<td>2.27</td>
<td>27682</td>
<td>4.37</td>
</tr>
<tr>
<td>kroB150</td>
<td>26130</td>
<td>26178</td>
<td>0.18</td>
<td>27909</td>
<td>6.81</td>
</tr>
<tr>
<td>d198</td>
<td>16780</td>
<td>15909</td>
<td>0.82</td>
<td>17397</td>
<td>10.25</td>
</tr>
<tr>
<td>kroA200</td>
<td>29368</td>
<td>29487</td>
<td>0.41</td>
<td>34087</td>
<td>16.07</td>
</tr>
<tr>
<td>kroB200</td>
<td>29437</td>
<td>30121</td>
<td>2.32</td>
<td>36980</td>
<td>25.62</td>
</tr>
<tr>
<td>a280</td>
<td>2579</td>
<td>2669</td>
<td>3.49</td>
<td>3157</td>
<td>22.41</td>
</tr>
<tr>
<td>Lin318</td>
<td>42029</td>
<td>43123</td>
<td>2.60</td>
<td>52156</td>
<td>24.10</td>
</tr>
<tr>
<td>pcb442</td>
<td>50778</td>
<td>52025</td>
<td>2.46</td>
<td>61979</td>
<td>22.06</td>
</tr>
</tbody>
</table>

| STD Deviation | 1.11 | 9.15 | 9.13 | 0.70 |

Case KROA-150    Case LIN-318
### Observations of Results

<table>
<thead>
<tr>
<th>Name</th>
<th>Bound</th>
<th>HASTS-ED</th>
<th>HASTS-CC</th>
<th>HASTS-CCED</th>
<th>HASTS-IEEA</th>
</tr>
</thead>
<tbody>
<tr>
<td>Att48</td>
<td>10628</td>
<td>10628</td>
<td>10653</td>
<td>10628</td>
<td>10628</td>
</tr>
<tr>
<td>eil51</td>
<td>426</td>
<td>426</td>
<td>426</td>
<td>426</td>
<td>426</td>
</tr>
<tr>
<td>Pr76</td>
<td>108159</td>
<td>108159</td>
<td>108159</td>
<td>108159</td>
<td>108159</td>
</tr>
<tr>
<td>kroA100</td>
<td>21282</td>
<td>21282</td>
<td>21282</td>
<td>21292</td>
<td>21282</td>
</tr>
<tr>
<td>kroB100</td>
<td>22141</td>
<td>22210</td>
<td>22200</td>
<td>22271</td>
<td>22141</td>
</tr>
<tr>
<td>will101</td>
<td>629</td>
<td>629</td>
<td>629</td>
<td>629</td>
<td>629</td>
</tr>
<tr>
<td>chl30</td>
<td>6110</td>
<td>6128</td>
<td>6150</td>
<td>6113</td>
<td>6113</td>
</tr>
<tr>
<td>kroA150</td>
<td>26524</td>
<td>26767</td>
<td>26727</td>
<td>26762</td>
<td>26525</td>
</tr>
<tr>
<td>kroB150</td>
<td>26130</td>
<td>26152</td>
<td>26860</td>
<td>26391</td>
<td>26130</td>
</tr>
<tr>
<td>d198</td>
<td>16780</td>
<td>16876</td>
<td>15796</td>
<td>15799</td>
<td>15781</td>
</tr>
<tr>
<td>kroA200</td>
<td>29368</td>
<td>29668</td>
<td>29487</td>
<td>29603</td>
<td>29479</td>
</tr>
<tr>
<td>kroB200</td>
<td>29437</td>
<td>30121</td>
<td>30121</td>
<td>30121</td>
<td>29543</td>
</tr>
<tr>
<td>a280</td>
<td>2579</td>
<td>2658</td>
<td>2669</td>
<td>2654</td>
<td>2579</td>
</tr>
<tr>
<td>lin318</td>
<td>42029</td>
<td>42938</td>
<td>43123</td>
<td>43083</td>
<td>42665</td>
</tr>
<tr>
<td>pcb442</td>
<td>50778</td>
<td>51860</td>
<td>52025</td>
<td>51955</td>
<td>51654</td>
</tr>
<tr>
<td>STD Deviation</td>
<td>1.05</td>
<td>1.26</td>
<td>1.07</td>
<td><strong>0.56</strong></td>
<td></td>
</tr>
</tbody>
</table>
1. We present a wide discussion on the current state-of-art meta-heuristics and their techniques.

2. We present a novel approach of characterizing different meta-heuristics into their common behavior, which consequently enables codes reuse across different meta-heuristics.

3. We describe the design and realization on how meta-heuristics can adopt a Request and Response (R&R) model that facilitates the formation hybridized schemes and related strategies.
QUESTIONS
SUPPLEMENTARY MATERIALS
Pure TS

- Implement *Strict Tabu Search*
  - Static Tabu tenure

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Solution:</strong></td>
<td>Single-dimension integer array</td>
</tr>
<tr>
<td><strong>Move:</strong></td>
<td>Swap-edge operator</td>
</tr>
<tr>
<td><strong>Constraint:</strong></td>
<td>Not applicable</td>
</tr>
<tr>
<td><strong>Neighborhood:</strong></td>
<td>( \binom{N}{2} ) pairs</td>
</tr>
<tr>
<td><strong>Objective:</strong></td>
<td>Sum of distance</td>
</tr>
<tr>
<td><strong>Penalty:</strong></td>
<td>Not applicable</td>
</tr>
<tr>
<td><strong>Tabu List:</strong></td>
<td>Tabu the “Moves” made</td>
</tr>
<tr>
<td><strong>Aspiration:</strong></td>
<td>Best-ever aspiration</td>
</tr>
</tbody>
</table>
## Pure ACO

- Implement **Ants Colony System**
  - Incorporate Exploitation and Exploration factor $q_0$
  - Update trails with the Iteration-best ants

<table>
<thead>
<tr>
<th><strong>Solution:</strong></th>
<th>Single-dimension integer array</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Move:</strong></td>
<td>Incremental operator</td>
</tr>
<tr>
<td><strong>Constraint:</strong></td>
<td>Not applicable</td>
</tr>
<tr>
<td><strong>Neighborhood:</strong></td>
<td>List of Unvisited Nodes</td>
</tr>
<tr>
<td><strong>Objective:</strong></td>
<td>Sum of distance</td>
</tr>
<tr>
<td><strong>Penalty:</strong></td>
<td>Not applicable</td>
</tr>
</tbody>
</table>

- **Local Heuristic:** $1 / \text{distance}$
- **Pheromone:** On arcs of every city
**HASTS SCHEMES**

**HASTS-EA (Empowered Ants)**

- **Inspired from**
  - *Pheromone trail*: Preference Table for intensification
  - *Tabu List*: Forbidden List for diversification

- **Design:**
  - Embed Tabu List into ACO Neighborhood generator
  - Tabu List prevents ants in the same iteration to construct same solutions
  - Encourage both intensification and diversification
  - Useful when there are many local optimal

- **Events**
  - None

- **Handlers**
  - None
HASTS SCHEMES

HASTS-IE (Improved Exploitation)

- Inspired from
  - ACO suffer from “crossing” of edges in the tour
  - “Over deposition” of pheromone

- Design:
  - TS removes “crossing” of edges from the tour
  - Apply TS at the end of ACO iterations
  - Improve Exploitation

- Events
  - End of ACO iterations before the pheromone update

- Handlers
  - Apply TS on the iteration-best solution
HASTS SCHEMES

HASTS-ED (Enhanced Diversification)

- Inspired from
  - TS suffer from cycling
  - Prominent with static tenure

- Design:
  - ACO as a probabilistic diversifier
    - Randomly destroy sub-routes for reconstruction
  - Apply ACO to TS if $n$ non-improving solutions are encountered
  - Enhance diversification

- Events
  - TS made $n$ number of non-improving moves

- Handlers
  - Apply probabilistic diversification on best-found solution
**HASTS SCHEMES**

**HASTS-CC (Collaborative Coalition)**

- **Inspired from**
  - ACO is an excellent constructing heuristic
  - TS is an excellent optimizing heuristic

- **Design:**
  - Two phase approach
    - ACO constructs a initial solution
    - TS then optimizes on the solution
  - Apply TS when all of ACO iterations are completed

- **Events**
  - ACO Engine stopped

- **Handlers**
  - Apply TS on ACO best-found solution
Two naïve hyper-hybrid schemes

**HASTS-IEEA Hyper-hybrid**
- Combine HASTS-IE with HASTS-EA
- Fuses tabu list from HASTS-EA into HASTS-IE
- Combined hybrid has a more aggressive diversifying capability

**HASTS-CCED Hyper-hybrid**
- Combine HASTS-ED with HASTS-CC
- TS of HASTS-CC is replaced with HASTS-ED
- Combined hybrid has enhanced optimizing phase
OBSERVATIONS OF RESULTS

Case Examination: KROA-150

- PURE TS
- PURE ACO
- HASTS IE
- HASTS EA
- HASTS IEEA

Graphs showing the results over time with different markers and lines for each category.
Observations of Results

Case Examination: LIN-318