Position of the work

- One EA key point :
  Exploitation / Exploration tradeoff

- One main practical difficulty :
  Choose operators, design, value of parameters, representation

- Parameters setting (Lobo et al. 2007) :
  - Off-line before the run : parameter tuning,
  - On-line during the run : parameter control.

- Increasing number of computation resources (GPU, ...)
  but, add parameters
Position of the work

- One EA key point: 
  Exploitation / Exploration tradeoff

- One main practical difficulty: 
  Choose operators, design, value of parameters, representation

- Parameters setting (Lobo et al. 2007): 
  - Off-line before the run: *parameter tuning*
  - On-line during the run: *parameter control*

- Increasing number of computation resources (GPU, ...)
  but, add parameters

---

**Distributed Adaptive Metaheuristic Selection (DAMS)**

- **Control** the execution of different *metaheuristics*
  in a **distributed environment**
Execution of metaheuristics in a distributed environment

- 3 metaheuristics \( \{M_1, M_2, M_3\} \)
- 4 nodes of computation

Which algorithm can we design?
Execution of metaheuristics in a distributed environment

- 3 metaheuristics \( \{M_1, M_2, M_3\} \)
- 4 nodes of computation

Which algorithm can we design?
Execution of metaheuristics in a distributed environment

- 3 metaheuristics \( \{M_1, M_2, M_3\} \)
- 4 nodes of computation

Which algorithm can we design?
Execution of metaheuristics in a distributed environment

- 3 metaheuristics \( \{M_1, M_2, M_3\} \)
- 4 nodes of computation
Execution of metaheuristics in a distributed environment

- 3 metaheuristics \( \{M_1, M_2, M_3\} \)
- 4 nodes of computation

Goal of control: Optimal strategy

The strategy that gives the best overall performances:
- At first, execution of \( M_3 \)
- then, \( M_1 \) and \( M_2 \), and then, \( M_1 \).
Point of view

- Distributed environment = unique system to control
- Attach a metaheuristic to each computational node
- Modify the topology if necessary

⇒ Toward heterogenous parallel EA (island model)
Point of view

- Distributed environment = unique system to control
- Attach a metaheuristic to each computational node
- Modify the topology if necessary

⇒ Toward heterogeneous parallel EA (island model)
Previous works

- **Fix strategy:**
  ex. 20% nodes 'exploratory' EA, 80% nodes 'exploitative' EA
Previous works

- **Fix strategy:**
  - ex. 20% nodes 'explorative' EA, 80% nodes 'exploitative' EA

- **Control of communication, topology:**
  - Gossip protocol, newcast protocol [Laredo et al., 10]
  - Heterogeneous island model: [Candan et al., 12]
    Migration rate control by the quality of operators

- **Control of parameters sharing global information:**
  - Master/worker to self-adapt pop. size [Bonnaire et al., 05]
  - Self-adapt pop. size, crossover [Srinivasa et al., 07]

- **Control of parameters, distributed environment:**
  - In each island: [Tongchim et al., 02]
    Testing 2 possible parameter settings on half population (EA)
    Send/receive (and mutate) the best parameters setting
Control : Local strategy

$m^n$ attachements with $n$ nodes and $m$ metaheuristics
**Control : Local strategy**

$m^n$ attachments with $n$ nodes and $m$ metaheuristics

**Local distributed strategy :**

For each computational node :

- Measure the efficiency of neighboring metaheuristics
- Select a metaheuristic according to the local information

Trade-off between :

*Exploitation* of the most promising metaheuristics

*Exploration* of new ones
Select Best and Mutate strategy (SBM)

For each nodes:

\[ k \leftarrow \text{initMeta()} \]
\[ P \leftarrow \text{initPop()} \]
\[ \text{reward} \leftarrow 0 \]
\[ \textbf{while Not stop do} \]
\[ \text{Migration of populations} \]
\[ \text{Send} \ (k, \text{reward}) \]
\[ \text{Receive} \ \forall i \ (k_i, \text{reward}_i) \]
\[ k \leftarrow \text{best meta from } k_i \text{’s} \]
\[ \textbf{if} \ \text{rnd}(0, 1) < p_{\text{mut}} \\textbf{then} \]
\[ k \leftarrow \text{random meta} \]
\[ \textbf{end if} \]
\[ P_{\text{new}} \leftarrow \text{meta}_k(P) \]
\[ \text{reward} \leftarrow \text{Reward}(P, P_{\text{new}}) \]
\[ P \leftarrow P_{\text{new}} \]
\[ \textbf{end while} \]
Select Best and Mutate strategy (SBM)

For each nodes:

\[ k \leftarrow \text{initMeta()} \]
\[ P \leftarrow \text{initPop()} \]
\[ \text{reward} \leftarrow 0 \]

\begin{algorithm}
while Not stop do
    Migration of populations
    Send \((k, \text{reward})\)
    Receive \(\forall i \ (k_i, \text{reward}_i)\)

    \[ k \leftarrow \text{best meta from } k_i \text{'s} \]
    \[ \text{if } \text{rnd}(0,1) < p_{mut} \text{ then} \]
    \[ k \leftarrow \text{random meta} \]
    \[ \text{end if} \]

    \[ P_{new} \leftarrow \text{meta}_k(P) \]
    \[ \text{reward} \leftarrow \text{Reward}(P, P_{new}) \]
    \[ P \leftarrow P_{new} \]
\end{algorithm}

end while
Experimental design

From 'Adaptive Operator Selection'

Dynamical Multi-Armed Bandit (DAMS) [Fiaho et al., GECCO 2008, 2009]

- 'Drosophilia' problems: OneMax problem \( (l = 10^4) \)
- Initial solution 000....0
- 4 mutations operators:
  - 5-bit, 3-bit, 1-bit, \( 1/l \) bit-flip
- \((1 + \lambda)\)–EA with \( \lambda = 50 \)

SBM-DAMS

- One iteration of \((1 + \lambda)\)–EA
- Reward: fitness gain \( f(x_{new}) - f(x) \)
- Three topologies: complete, circle, grid graph.
- \( p_{mut} = 10^{-3}, \{0.0005, 0.005, 0.001, 0.01, 0.05, 0.1, 0.2, 0.3\} \)
- 20 independent runs
Performances, adaptation properties

Grid topologies:

Circle topologies:

No distributed adaptive algorithm in the literature

Algorithms:
- Selection of 'good' meta important?
  ⇒ Random selection of metaheuristic (Rnd-DAMS)
- Optimal independent strategy?
  ⇒ Selection with sequential oracle (seqOracle-DAMS)

SBM outperforms (∀ topo., ∀ n):
  Random selection
  Sequential oracle
  ⇒ Adaptation of SBM very efficient
Dynamics of SBM-DAMS

Average frequency and fitness

Efficiency of local communication

For $n$ nodes around fitness 0:

1/1 bit-flip mutation $> 5$-bit mutation

For fitness 0,

$$\text{IP}(\text{gain bit-flip} > 5) = 1 - \alpha^{\lambda \cdot n}$$

$\alpha$ prob at most 5 bits are flipped

Whatever the topology:

- Decentralized decision (but not independent)
- Global property: correct mutation operator selection

First, 1/1 bit-flip mutation (opposite of sequential oracle)

Then 5-bits, 3-bits, 1-bit, 1/1 bit-flip
Robustness of metaheuristic mutation parameter $p_{mut}$

Grid topologies:

![Graph showing performance with varying mutation rates for grid topologies]

Circle topologies:

![Graph showing performance with varying mutation rates for circle topologies]

- Range of performances small according to metaheuristic mutation rate
- $p_{mut} \in [10^{-3}, 10^{-1}]$

⇒ Quite robust parameter for this problem
Parallel properties: SBM-DAMS vs. sequential AOS

- Sequential Adaptive Operator Selection:
  - DMAB running (1 + 50)-EA
- SBM-DAMS:
  - 50 nodes (complete graph) running (1 + 1)-EA

Performances (number of evaluations):
- oracle < ex-DMAB < SBM < DMAB

More information for AOS:
- DMAB: update operator every mutation
- SBM: update every $n$ mutations

But, SBM time could be divided by $n$
Distributed Adaptive Metaheuristic Selection
A generic framework

\[
M \leftarrow \text{INIT\_META()}
\]
\[
P \leftarrow \text{INIT\_POP()}
\]
\[
S \leftarrow \text{INIT\_STATE()}
\]
\[
I \leftarrow \{M, P, S\}
\]
repeat
Distributed Adaptive Metaheuristic Selection
A generic framework

\[ M \leftarrow \text{INIT\_META}() \]
\[ P \leftarrow \text{INIT\_POP}() \]
\[ S \leftarrow \text{INIT\_STATE}() \]
\[ I \leftarrow \{M, P, S\} \]
\[ \text{repeat} \]
\[ \quad /\!/ \text{Distributed Level} /\!\! / \]
\[ \quad c \leftarrow \text{LOCAL\_COMMUNICATION}(I) \]
\[ \quad P \leftarrow \text{UPDATE\_POPULATION}(I, c) \]
\[ \quad S \leftarrow \text{UPDATE\_LOCAL\_STATE}(I, c) \]
Distributed Adaptive Metaheuristic Selection
A generic framework

\[
\begin{align*}
M & \leftarrow \text{INIT\_META()} \\
P & \leftarrow \text{INIT\_POP()} \\
S & \leftarrow \text{INIT\_STATE()} \\
I & \leftarrow \{M, P, S\} \\
\text{repeat} & \\
\quad & \text{/** Distributed Level **/} \\
\quad & c \leftarrow \text{LOCAL\_COMMUNICATION}(I) \\
\quad & P \leftarrow \text{UPDATE\_POPULATION}(I, c) \\
\quad & S \leftarrow \text{UPDATE\_LOCAL\_STATE}(I, c) \\
\quad & \text{/** Metaheuristic Selection Level **/} \\
\quad & M \leftarrow \text{SELECT\_META}(I)
\end{align*}
\]
Distributed Adaptive Metaheuristic Selection
A generic framework

\[ M \leftarrow \text{INIT\_META}() \]
\[ P \leftarrow \text{INIT\_POP}() \]
\[ S \leftarrow \text{INIT\_STATE}() \]
\[ I \leftarrow \{M, P, S\} \]

\textbf{repeat}

\textbf{/** Distributed Level **/}
\[ c \leftarrow \text{LOCAL\_COMMUNICATION}(I) \]
\[ P \leftarrow \text{UPDATE\_POPULATION}(I, c) \]
\[ S \leftarrow \text{UPDATE\_LOCAL\_STATE}(I, c) \]

\textbf{/** Metaheuristic Selection Level **/}
\[ M \leftarrow \text{SELECT\_META}(I) \]

\textbf{/** Atomic Low Level **/}
\[ (P, S) \leftarrow \text{APPLY\_META}(M, P) \]

\textbf{until} \text{STOPPING\_CONDITION}(I)
Distributed Adaptive Metaheuristic Selection
A generic framework

\[
\begin{align*}
M & \leftarrow \text{INIT\_META}(\cdot) \\
P & \leftarrow \text{INIT\_POP}(\cdot) \\
S & \leftarrow \text{INIT\_STATE}(\cdot) \\
I & \leftarrow \{M, P, S\} \\
\text{repeat} & \\
\quad /\!\!\!\!\!\!\!\!\!/** \text{Distributed Level} **/ \\
\quad c & \leftarrow \text{LOCAL\_COMMUNICATION}(I) \\
\quad P & \leftarrow \text{UPDATE\_POPULATION}(I, c) \\
\quad S & \leftarrow \text{UPDATE\_LOCAL\_STATE}(I, c) \\
\quad /\!\!\!\!\!\!\!\!\!/** \text{Metaheuristic Selection Level} **/ \\
\quad M & \leftarrow \text{SELECT\_META}(I) \\
\quad /\!\!\!\!\!\!\!\!\!/** \text{Atomic Low Level} **/ \\
\quad (P, S) & \leftarrow \text{APPLY\_META}(M, P) \\
\text{until} & \text{STOPPING\_CONDITION}(I) \\
\end{align*}
\]

\[
\begin{align*}
k & \leftarrow \text{initMeta}(\cdot) \\
P & \leftarrow \text{initPop}(\cdot) \\
\text{reward} & \leftarrow 0 \\
\text{while} \text{ Not stop do} & \\
& \quad \text{Migration of populations} \\
& \quad \text{Send } (k, \text{ reward}) \\
& \quad \text{Receive } \forall i \ (k_i, \text{ reward}_i) \\
& \quad k \leftarrow \text{best meta from } k_i\text{'s} \\
& \quad \text{if } \text{rnd}(0, 1) < p_{\text{mut}} \text{ then} \\
& \quad \quad k \leftarrow \text{random meta} \\
& \quad \text{end if} \\
& \quad P_{\text{new}} \leftarrow \text{meta}_k(P) \\
& \quad \text{reward} \leftarrow \text{Reward}(P, P_{\text{new}}) \\
& \quad P \leftarrow P_{\text{new}} \\
\text{end while}
\end{align*}
\]
Distributed Adaptive Metaheuristic Selection
A generic framework

\[ M \leftarrow \text{INIT\_META}() \]
\[ P \leftarrow \text{INIT\_POP}() \]
\[ S \leftarrow \text{INIT\_STATE}() \]
\[ I \leftarrow \{M, P, S\} \]

repeat
  //** Distributed Level **/
  \[ c \leftarrow \text{LOCAL\_COMMUNICATION}(I) \]
  \[ P \leftarrow \text{UPDATE\_POPULATION}(I, c) \]
  \[ S \leftarrow \text{UPDATE\_LOCAL\_STATE}(I, c) \]

  //** Metaheuristic Selection Level **/
  \[ M \leftarrow \text{SELECT\_META}(I) \]

  //** Atomic Low Level **/
  \[(P, S) \leftarrow \text{APPLY\_META}(M, P)\]

until \text{STOPPING\_CONDITION}(I)

\[ k \leftarrow 0 \]
\[ P \leftarrow \text{initPop()} \]

while Not stop do
  Migration of populations
  \[ P_{\text{new}} \leftarrow \text{meta}_k(P) \]

  \[ P \leftarrow P_{\text{new}} \]
end while
Conclusions and Futur Works

- **On-line control** of metaheuristics execution in **distributed environment**
- 'Select Best and Mutate' strategy: efficient adaptation and parallel properties

**Future works**

- **Theoretical works** on running-time (in progress)
- Study the performances on NP-hard **multimodal problems**
- Use together **different metaheuristics** (EA, memetic EA, ant, etc.)
- Study addition and deletion of computational nodes
Future works

\[ M \leftarrow \text{INIT\_META()} \]
\[ P \leftarrow \text{INIT\_POP()} \]
\[ S \leftarrow \text{INIT\_STATE()} \]
\[ I \leftarrow \{M, P, S\} \]

repeat
  /** Distributed Level **/
  \[ c \leftarrow \text{LOCAL\_COMMUNICATION(I)} \]
  \[ P \leftarrow \text{UPDATE\_POPULATION(I, c)} \]
  \[ S \leftarrow \text{UPDATE\_LOCAL\_STATE(I, c)} \]

  /** Metaheuristic Selection Level **/
  \[ M \leftarrow \text{SELECT\_META(I)} \]

  /** Atomic Low Level **/
  \[ (P, S) \leftarrow \text{APPLY\_META}(M, P) \]

until \text{STOPPING\_CONDITION(I)}

Metaheuristic Selection Level :

- Design selection metaheuristic methods (UCB strategy)

Distributed Level :

- Rewards : cumulative measures
- Information, state :
  Communication of rewards
- Asynchronous method
- Binary rewards (crossover)
- Define an adaptive migration policy

Environnement Level :

- grid5000