Towards a resolution of the firing squad problem with 5 states by metaheuristics

FRAC 2007 - Nice

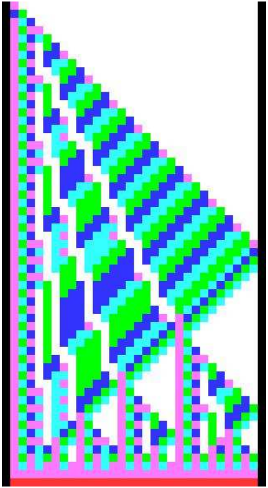
M. Frandeș S. Verel M. Clergue P. Collard

Laboratoire I3S - Équipe TEA - Université de Nice-Sophia Antipolis / CNRS

29 juin 2007

Outline

Firing squad problem (J.Myhill 1957)



- one firing man = one cell
- line of length n of firing squad
- Initial configuration :
  - the left cell in "general" state
  - right cells in "quiescent" state
- Exchange information between neighbors firing men = local transition function
- Goal : find a transition function such that
  $\forall n$, the final configuration is a line of cells in "firing" state.

Previous works

Transition function which solve FSP in optimal time ($2n - 2$ iterations) :

- E. Goto [1962] with thousand of states
- Waksman [1966] with 16 states
- Balzer [1967] with 8 states
- Mazoyer [1987] with 6 states

No transition function with 4 states (Balzer)

Previous works

Transition function which solve FSP in optimal time ($2n - 2$ iterations) :

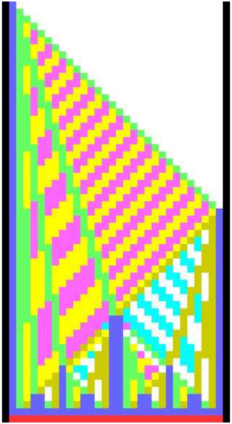
- E. Goto [1962] with thousand of states
- Waksman [1966] with 16 states
- Balzer [1967] with 8 states
- Mazoyer [1987] with 6 states

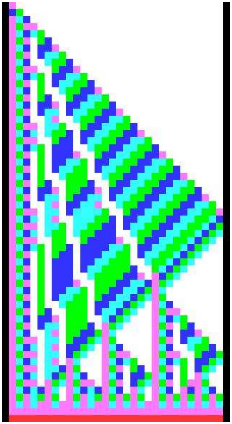
No transition function with 4 states (Balzer)

Open Problem :

Does it exist transition which solve FSP with 5 states ?

Space-time diagram at Balzer rules at 8 états (left) and of Mazoyer at 6 states (right)





Goal of the work

- Goal : solve the firing squad problem (FSP) at 5 states

New approach to try for solving FSP :

- Definition of a family of optimization problems such that : Solving the family give a solution of FSP
- Solving the optimization problems (large and unknow search space) by the mean of stochatics optimization : metaheuristics

Motivations of this work

- Enrico Formenti : “je connais un problème que tu pourrais essayer avec tes trucs”
- Previous works on Majority problem
  (find transition which classify the configuration according to density) :
  - analyze of the search space
  - design a new optimization algorithm
- objectif function is “black box” :
  no derivation, no analytic expression of the function
- Design new stochatic algorithms for hard optimization (NP-complet or more)
- example of automatic programmation of parallel machines

Methodology

Goal : solve the firing squad problem at 5 states by the mean of stochatic optimization

1. Define two optimization problems :
    - MAXF_n : for cell's line of length n, optimization fonction is the number of synchronised cells in "firing" state
    - $\mathrm{MAXF}_{2,n}$: optimization fonction is the sum of the number of synchronised cells in "firing" state for all line which the length is between 2 and n

- Solving $\forall n \geq 2, \mathrm{MAXF}_{2,n}$ to solve the firing squad problem

2. Use and design metaheuristics to solve the optimisation problems

Finite linear cellular automata (flCA)

$\mathcal{A} = (\mathcal{Q}, \delta)$

- $\mathcal{Q}$ finite set of states
- $q_l,\ q_i,\ q_f \in \mathcal{Q}$
- δ local transition function
- $\delta : \mathcal{Q} \cup q_b \times \mathcal{Q} \times \mathcal{Q} \cup q_b \rightarrow \mathcal{Q}$
- $\delta(q_b, q_l, q_l) = \delta(q_l, q_l, q_l) = \delta(q_l, q_l, q_b) = q_l$

Configurations :

- initial : $\mathcal{C}_0(1) = q_i$ and $\forall j \in]1, n]$, $\mathcal{C}_0(j) = q_l$
- final : $\forall i \in [1, n], \mathcal{C}_s(i) = q_f$

Optimisation problem

Definition

$\mathcal{P}=(\mathcal{S}, f)$

- $\mathcal{S}$: set of potential solutions, search space
- $f : \mathcal{S} \rightarrow \mathbb{R}$: objective function to maximize (or minimize)

Goal : find the set $\mathcal{S}_{opt}$ such that

$$\forall s_{opt} \in \mathcal{S}_{opt}, \forall s \in \mathcal{S}, f(s) \leq f(s_{opt})$$

...or find solutions which is enough good for the “user” of the problem

In the firing squad problem,
a solution = a transition function (set of rules)

Metaheuristics

Heuristic : solving algorithm based on the "experience" which not necessary give an optimal solution

but I'd like :

- the most often as possible, a solution near optimal solutions
- the less often, a bad solution
- with low complexity
- simple to compute (in light version...)

Metaheuristics

Heuristic : solving algorithm based on the "experience" which not necessary give an optimal solution

but I'd like :

- the most often as possible, a solution near optimal solutions
- the less often, a bad solution
- with low complexity
- simple to compute (in light version...)

Metaheuristic : set of heuristic defined by

- parameters which describe the set of heuristics
- a description of the heuristic in general term

Metaheuristics classification

Stochatic algorithms with unique solution :

- (Random search),
- Ascent Algorithms : Hill-Climber (HC), first-ascent
- Simulated Annealing (Kirkpatrick *et al* 1983)
- Tabu search (Glover 1986)

Stochatic algorithms with population of solutions :

- Genetic algorithms (Holland 1975 and even before) : binary strings
- Evolution strategies (Schweffel 1970) : real optimization
- Genetic programming (Koza 92) : "program"
- Evolutionary Algorithm : the oecumenic algorithm
- Ant optimization (Bonabeau 1999) : route on graph
- memetic algorithms : combined local search with population

Stochatic algorithms with unique solution

3 parts of local search algorithms :

- $\mathcal{S}$ set of solutions (search space),
- $f : \mathcal{S} \rightarrow \mathbb{R}$ objective function
- $\mathcal{V}(s)$ neighborhood, set of neighbor solutions of s

Generic algorithm

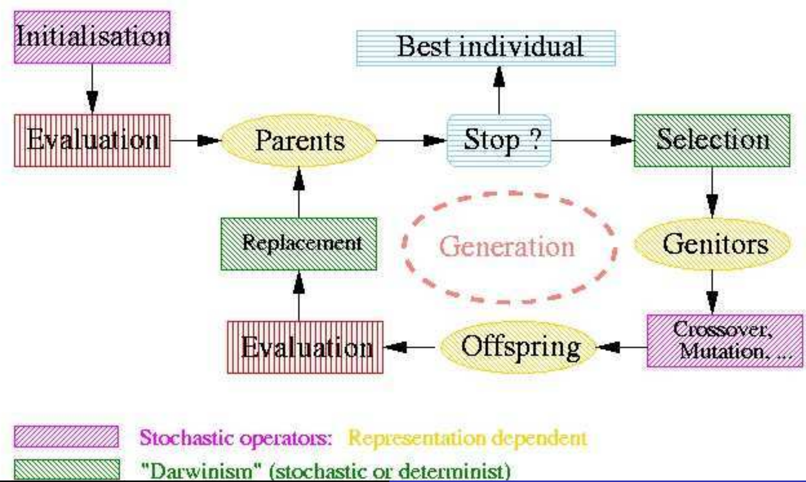
Choose initial solution $s \in \mathcal{S}$
repeat
  choose $s^{'} \in \mathcal{V}(s)$
  $s \leftarrow s^{'}$
until stop criterium verify

Ascent algorithms

Hill-Climbing	First-Ascent

Choose randomly initial solution $s \in \mathcal{S}$
repeat
    Choose $s' \in \mathcal{V}(s)$ such as $f(s')$ is maximal
    $s \leftarrow s'$
until s local optimum
	step $\leftarrow 0$
	Choose randomly initial solution $s \in \mathcal{S}$
	repeat
	    choose randomly $s' \in \mathcal{V}(s)$
	    if $f(s) \leq f(s')$ **then**
	        $s \leftarrow s'$
	    end if
	    step $\leftarrow$ step $+1$
	until stepMax $\leq$ step

Generic Evolutionary Algorithm



Memetic algorithms

Combined :

- efficient local search (local operator)
- population search based (global operator)

⇒ initialisation of the population use local search
⇒ mutation operator is replaced by local search

Avantages :

- at least, the same performance as efficient local search !
- better compromize between exploration/exploitation

MAXF_n and $\mathrm{MAXF}_{2,n}$ families of problems

- Optimization problem MAXF_n :

$$f_n : \mathcal{S} \to [0, n]$$

$$f_n(s) = \begin{cases} 0 & \text{if } \forall t \leq T_{max},\ \omega(C_t) = 0 \\ \omega(C_t) & \text{if } \exists t \leq T_{max},\ \forall t' < t,\ \omega(C_{t'}) = 0 \end{cases}$$

T_{max} is the limit time before synchronisation
ω counts the number of firing state in the configuration C_t
$\omega(C_t) = \sum_{i=1}^{n} \xi^{q_f}_{C_t(i)}$

- Optimization problem $\mathrm{MAXF}_{2,n}$:

$$f_{2,n} : \mathcal{S} \to [0, \frac{n(n+1)}{2} - 1]$$

$$f_{2,n}(s) = \sum_{i=2}^{n} f_i(s)$$

Search space

$\mathcal{S} = \{ \ \delta \mid (\mathcal{Q}, \delta) lfAC, \ \sharp\mathcal{Q} = 5 \}$, but rules with "firing" state are not use

Neighborhood of s : solutions with one rule changed

- Whole search space :
  - number of rules : $4^3 - 1$ "central" rules, $4^2 - 1$ rules for left and right bounds
    So, 93 rules
  - size of search space : 5 possible states for each rule
    $5^{93} \approx 10^{66}$
  - size of neighborhood : 4 other possible states for each rule
    $93 \times 4 = 372$

Search space : Restricted search space

$\mathcal{S} = \{ \ \delta \mid (\mathcal{Q}, \delta) \mathit{lfAC}, \ \sharp\mathcal{Q} = 5\}$, but rules with "firing" state are not use

Neighborhood of s : solutions with one rule changed

- Only "firing" state for the rules $333 \rightarrow q_f$ and $33 \rightarrow q_f$
  - number of rules : 90 rules
  - size of search space : 4 possible states for each rule $4^90 \approx 10^{54}$
  - size of neighborhood : 3 other possible states for each rule $90 \times 4 = 270$

Search space : Restricted search space

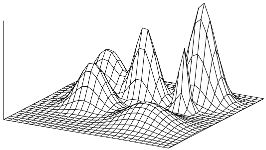
$\mathcal{S} = \{ \ \delta \mid (\mathcal{Q}, \delta) \mathit{lfAC}, \ \sharp\mathcal{Q} = 5\}$, but rules with "firing" state are not use

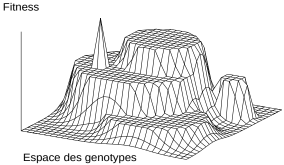
Neighborhood of s : solutions with one rule changed

- Only "firing" state for the rules $333 \rightarrow q_f$ and $33 \rightarrow q_f$
  - number of rules : 90 rules
  - size of search space : 4 possible states for each rule $4^90 \approx 10^{54}$
  - size of neighborhood : 3 other possible states for each rule $90 \times 4 = 270$

In which search space is it easier to find good solutions ?

Fitness landscape





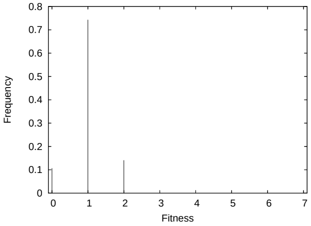
Fitness landscape $(\mathcal{S}, \mathcal{V}, f)$:

- $\mathcal{S}$: set of potential solutions,
- $\mathcal{V} : \mathcal{S} \rightarrow 2^{\mathcal{S}}$: neighborhood relation,
- $f : \mathcal{S} \rightarrow \mathbb{R}$: objective function.

Fitness landscape

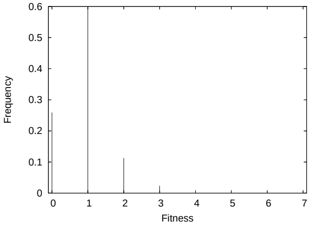
Frequency of the value of objective function for random solutions

Whole search space



average $1.0522_{0.531107}$
maximum 5

Restrited search space



average $0.9264_{0.739042}$
maximum 7

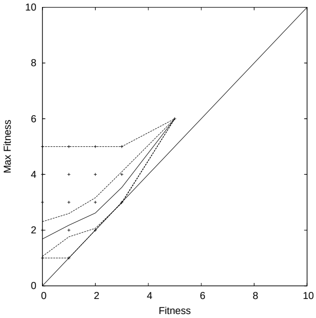
- Average is lower for Restrited SS (more 0 fitness value)
- The tail is higher for restrited SS

⇒ Restrited seems "easier"

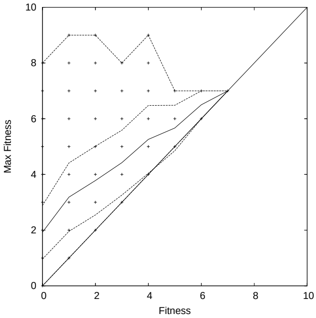
Fitness landscape

Correlation between high fitness value in the neighborhood with fitness value of solutions

Whole search space



restrited search space

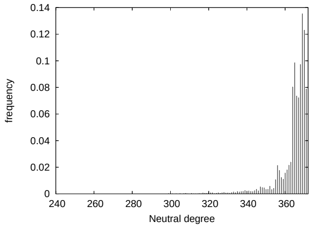


$\Rightarrow$ Restrited seems "easier"

Fitness landscape

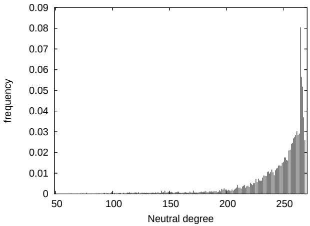
Number of equal fitness value in the neighborhood (Neutrality)

Whole search space



$363.926 \approx 97.8\%$

restrited search space



$243.865 \approx 90\%$

$\Rightarrow$ Restrited seems "easier"

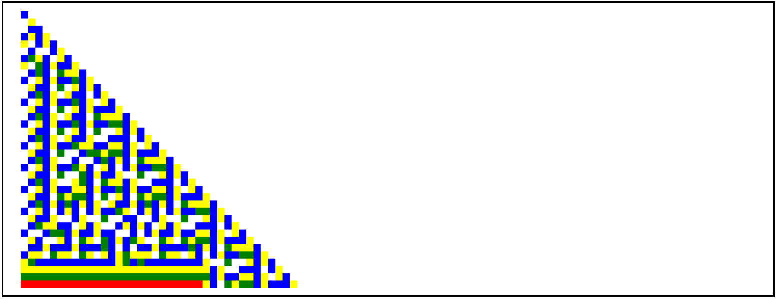
Results on MAXF_{101}

Metaheuristic	$mean_{std.dev.}$	Max.
HC	$3.7871_{1.9767}$	18
First-Ascent	$13.735_{2.73067}$	25
AE	$6.2471_{1.03071}$	11

Results of memetic algorithm on MAXF_{101}

	Local search rate				
	0.1	0.5	0.8	0.9	1.0
0.0	$13.8_{2.1}$ 18	$15.9_{3.4}$ 27	$16.5_{2.6}$ 25	$16.3_{2.6}$ 23	$16.3_{2.4}$ 23
0.3	$13.7_{2.7}$ 20	$16.3_{2.7}$ 25	$15.9_{2.1}$ 20	$16.7_{2.3}$ 22	$17.8_{3.1}$ 26
0.5	$14.3_{2.6}$ 21	$17.2_{3.7}$ 29	$16.7_{2.0}$ 20	$17.6_{2.3}$ 22	$17.0_{2.6}$ 22
0.8	$15.2_{2.7}$ 22	$16.2_{2.4}$ 22	$17.5_{2.6}$ 23	$17.5_{2.2}$ 23	$17.4_{2.8}$ 26
1.0	$14.2_{1.7}$ 17	$16.5_{2.4}$ 21	$17.1_{2.3}$ 24	$16.6_{2.4}$ 25	$16.6_{2.7}$ 25

Space-time diagram of best transition function on MAXF_{101}

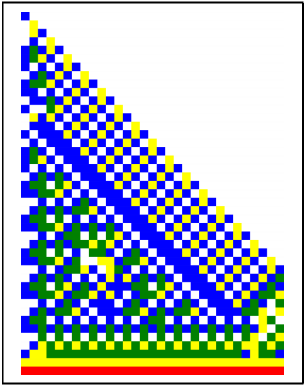


31 cells synchronized in firing state

Solved problems of MAXF_n

Metaheuristic	Solved length n
Hill-Climbing	$2, 3, 4, \ldots, 20$
First-Ascent	$2, 3, 4, \ldots, 29$
memetic-FA	$2, 3, 4, \ldots, 31$

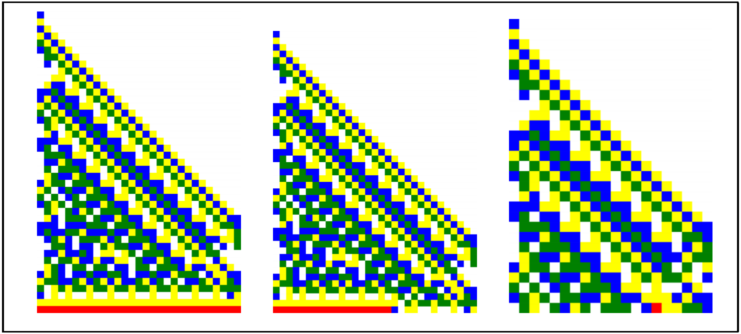
Space-time diagram of trans. func. which solve MAXF_{31}



31 men which open fire in the same time !

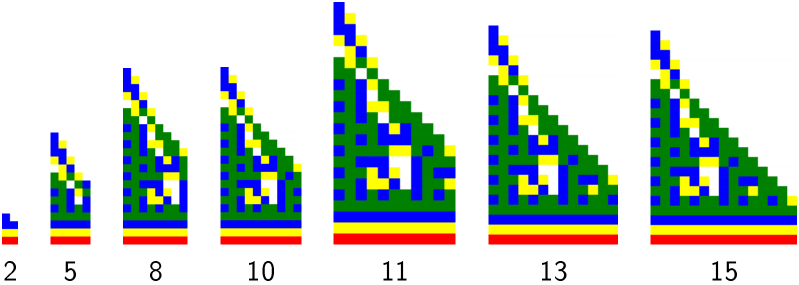
Space-time diagram of the same solution for different lengths

But only a specific length is solved.



Results on $\mathrm{MAXF}_{2,n}$

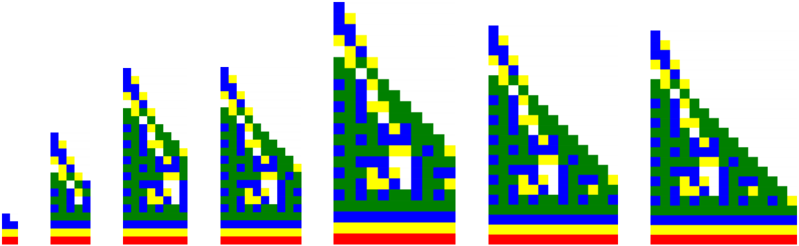
memetic-FA : Solve $\forall n \in \{2, 3, 4, \ldots, 15\}$



Space-time diagram for a solution for the length 15

Results on $\mathrm{MAXF}_{2,n}$

memetic-FA : Solve $\forall n \in \{2, 3, 4, \ldots, 15\}$



2 5 8 10 11 13 15

Space-time diagram for a solution for the length 15

...do not synchronize 16 cells

Comparison with the result of Balzer

With 4 states :

- Balzer program shows the non existence of a transition function
- Find transition function which synhcronise $\forall n \in \{2, \dots, 8\}$, but not $\forall n \in \{2, \dots, 8, 9\}$

With 5 states

- $\exists$ transition function which synchronise $\forall n \in \{2, \dots, 15\}$
- So, for the same prove, we must show for length > 15

Conclusion

- our approach : stochatic optimization based on local search and population (metaheuristics)
- define families of optimization problems
- transition functions is specific for cells line of length ≤ 31
- Find some transition functions which synchronise line which havs a length between 2 and 15
- FSP is very hard optimization problem ... still an open question

Perspectives

- Explain the space-time diagram of the transition functions (signals,...) to design more efficient algorithms
- Define new objective function (take care of the synchronising time,...)
- Define some middle stages
- Uses some other metaheuristic (simulated annealing, tabu search,...)
- Design a efficient recombinaison between solutions (based on frequency of use of rules,...)
- Study the search space
- Study the search spaceS

Define new objective function