Divide-and-Evolve

An Evolutionary Metaheuristic for Domain-Independent Satisficing Planning

Jacques Bibai[1,2] , Pierre Savéant[2], Marc Schoenauer[1], Vincent Vidal[3]

7th Annual HUMIES Awards

GECCO 2010

[1] Project-team TAO, INRIA Saclay Île-de-France

[2] Thales Research & Technology, Palaiseau, France

[3] CRIL, Université d'Artois, Lens, now with ONERA DCSD, Toulouse, France

AI Planning problem

Input : <A, O, I, G>

- Domain:

  A: set of atoms, O: set of actions

- Instance:

  I: initial state, G: Goal state

Output : Optimal Plan

Ordered set of actions: when executed in state I, leads to a state where G is satisfied

Quality :

- Classic problems (aka STRIPS): Number of actions
- Actions with cost: Total cost
- Temporal planning: makespan (total duration)
  actions can be run in parallel

Complexity :

PSPACE-complete for classical planning [Bylander 1994]

EXPSPACE-complete for temporal planning [Rintanen 2007]

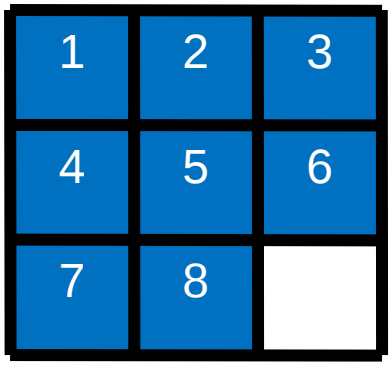
PDDL representation model

- Developed for the International Planning Competition (IPC) series
- Allows to compare different planners on given benchmarks problems

Exemple

Domain: Mystic_puzzle

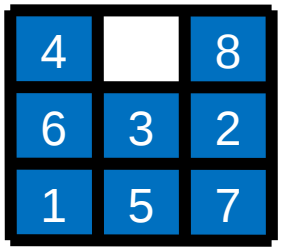
```
(define (domain MYSTIC_PUZZLE)
(:types position tile)
(:predicates (at ?tile - tile ?position - position)
             (neighbor ?p1 - position ?p2 - position)
             (empty ?position – position))
(:action move
    :parameters (?tile - tile ?from ?to - position)
    :precondition (and (neighbor ?from ?to)
                       (at ?tile ?from)
                       (empty ?to))
    :effect (and (at ?tile ?to)
                 (empty ?from)
                 (not (at ?tile ?from))
                 (not (empty ?to))))
)
```

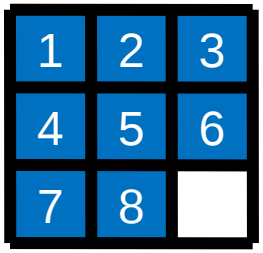


Instance

```
(define (problem MYSTIC_PUZZLE-3x3)
  (:domain MYSTIC_PUZZLE)
  (:objects
     p_1_1 p_1_2 p_1_3 p_2_1 p_2_2 p_2_3 p_3_1 p_3_2 p_3_3 - position
     t_1 t_2 t_3 t_4 t_5 t_6 t_7 t_8 - tile)
  (:init
    (neighbor p_1_1 p_1_2) (neighbor p_1_2 p_1_1) (neighbor p_1_2 p_1_3)
    (neighbor p_1_3 p_1_2) (neighbor p_2_1 p_2_2) (neighbor p_2_2 p_2_1)
    (neighbor p_2_2 p_2_3) (neighbor p_2_3 p_2_2) (neighbor p_3_1 p_3_2)
    (neighbor p_3_2 p_3_1) (neighbor p_3_2 p_3_3) (neighbor p_3_3 p_3_2)
    (neighbor p_1_1 p_2_1) (neighbor p_2_1 p_1_1) (neighbor p_1_2 p_2_2)
    (neighbor p_2_2 p_1_2) (neighbor p_1_3 p_2_3) (neighbor p_2_3 p_1_3)
    (neighbor p_2_1 p_3_1) (neighbor p_3_1 p_2_1) (neighbor p_2_2 p_3_2)
    (neighbor p_3_2 p_2_2) (neighbor p_2_3 p_3_3) (neighbor p_3_3 p_2_3)
    (at t_4 p_1_1) (empty p_1_2) (at t_8 p_1_3) (at t_6 p_2_1) (at t_3 p_2_2)
    (at t_2 p_2_3) (at t_1 p_3_1) (at t_5 p_3_2) (at t_7 p_3_3))

(:goal
    (at t_1 p_1_1) (at t_2 p_1_2) (at t_3 p_1_3) (at t_4 p_2_1) (at t_5 p_2_2)
    (at t_6 p_2_3) (at t_7 p_3_1) (at t_8 p_3_2))
```





State of the art (man-made) planners

Best performing (from IPC 2008 and more recent publications)

- LPG [Gerevini, Saetti, and Serina, 2004]: classical and temporal planning
  Stochastic Local search approach and temporal action graphs
- LAMA [Richter and Westphal, 2009]: classical and cost planning
  Fast Downward with FF heuristic + landmark heuristic + iterated WA*
- Temporal Fast Downward [Röger, Eyerich, and Mattmüller, 2009]: temporal planning
  Temporal/numeric extension of Fast Downward; searches time-stamped states

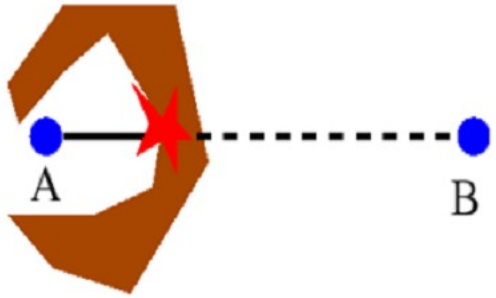
Useful 'extreme' planners

- CPT [Vidal 2006], classical, cost and temporal planning
  Partial order causal links + constraint programming
  Optimal, but far too slow (fails on most medium to large instances)
- YAHSP [Vidal, 2004], classical, cost and temporal planning
  Lookahead strategy planning system (no backtrack)
  Fast and robust, but very poor quality

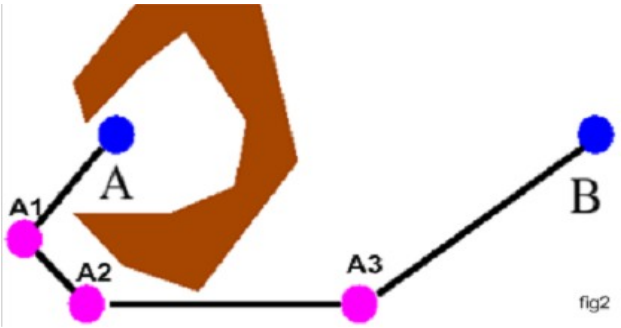
Motivation

- Given a 'local' ('weak' or 'greedy') algorithm
- Solve problem this local algorithm cannot solve
- Improve quality of solutions of the weak/greedy algorithm

Rationale: evolutionary sequential Divide-and-Conquer



Local algorithm fails



Local algorithm can solve the sequence of easy problems

Schoenauer, M. , Savéant, P. and Vidal, V.. Divide-and-Evolve: a New Memetic Scheme for Domain-Independent Temporal Planning. In J. Gottlieb and G. Raidl, eds., EvoCOP'06, LNCS 3906, pp. 247-260, Springer Verlag, 2006.

Schoenauer, M., Savéant, P., Vidal, V.: Divide-and-Evolve: a Sequential Hybridisation Strategy using Evolutionary Algorithms. In: Z. Michalewicz and P. Siarry eds., Advances in Metaheuristics for Hard Optimisation, pp. 179–198. Springer Verlag, 2007.

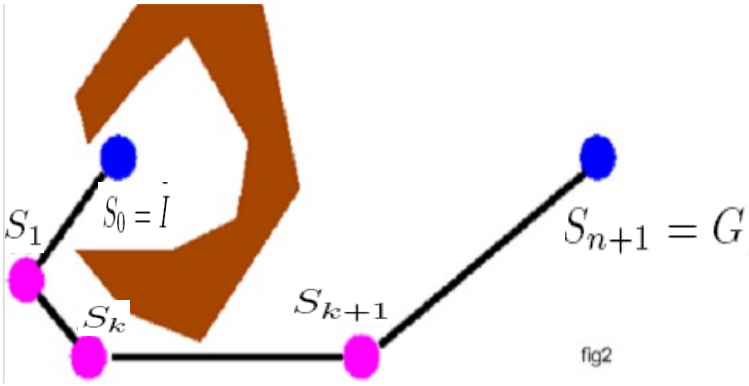
Problem

<A, O, I, G> = $\mathcal{P}_{\mathcal{D}}$(I,G)

Representation

Ordered list of partial states

S_0=I, S_1, ..., S_n, S_{n+1} =G



Evaluation

Solve consecutive sub-problems $\mathcal{P}_{\mathcal{D}}(S_k,S_{k+1})$, k∈[0,n]

with 'greedy' planner CPT or 'weak' planner YAHSP

Fitness

- All problems solved: sum of plan qualities
- Fails solving $\mathcal{P}_{\mathcal{D}}(S_l, S_{l+1})$: penalty + “dist”($S_l$,G)

State representation

Completeness vs size of search space: which predicates to use?

From expert manual choice to statistic-driven stochastic sampling

Consistency: No semantics in PDDL → how to avoid (at t_4 p_1_1) and (at t_4 p_1_2)?

Use pairwise mutex from planner (CPT/YAHSP)

Initialization and variation operators

Blind to fitness, not to domain knowledge

Use reachability heuristics [Haslum & Geffner 2000] to restrict the choices

An intricate memeticization

Schoenauer, M. , Savéant, P. and Vidal, V.. Divide-and-Evolve: a New Memetic Scheme for Domain-Independent Temporal Planning. In J. Gottlieb and G. Raidl, eds.: EvoCOP'06, LNCS 3906, pp. 247-260, Springer Verlag, 2006.

Bibai, J. , Savéant, P. , Schoenauer, M. and Vidal, V.. An Evolutionary Metaheuristic Based on State Decomposition for Domain-Independent Satisficing Planning. In R. Brafman et al., eds, Proc. ICAPS 2010, pp 15-25, AAAI Press, 2010.

Experimental settings

- (100+700)-ES evolution engine
  100 parents, 700 offspring, next parents are the best of the 800 (parents+offspring)

- Stop if no improvement after 50 generations

  Maximum of 1000 generations (or 30mn CPU for IPC rules)

- 1-point Crossover: 20%

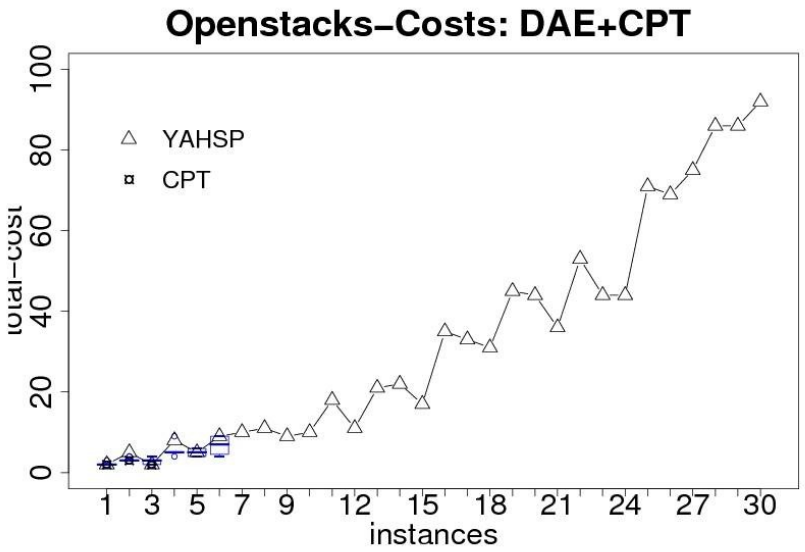
- Mutation: 80%, with relative weights
  - 50% add state mutation
  - 16.66% delete state mutation
  - 16.66% add/change atoms mutation
  - 16.66% delete atoms mutation

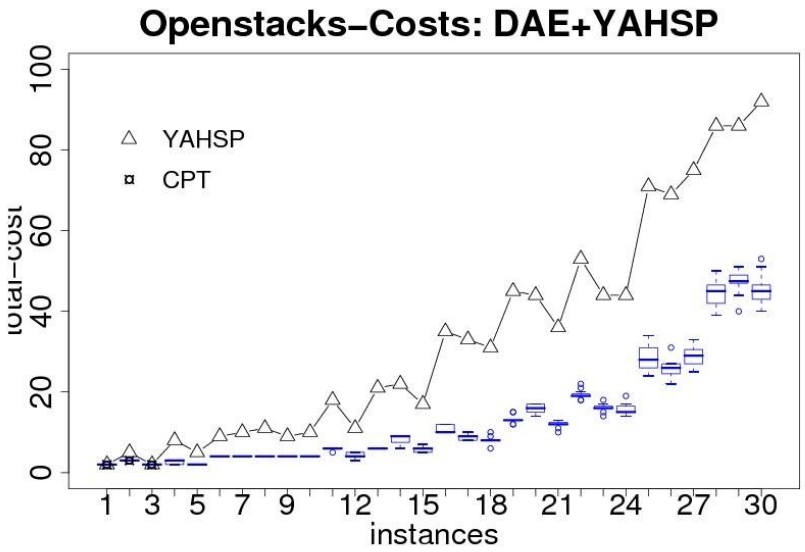
Bibai, J.; Savéant, P.; Schoenauer, M.; and Vidal, V. Learning Divide-and-Evolve Parameter Configurations with Racing (ICAPS 2009 – Workshop on Planning and Learning)

See also

Bibai, J. , Savéant, P. , Schoenauer, M. and Vidal, V.. On the Generality of Parameter Tuning in Evolutionary Planning. In GECCO 2010

Greedy vs weak embedded planner





Results

- DAE+X better than X alone (X=CPT or YAHSP)
- DAE+YAHSP much better than DAE+CPT
- Solutions very close to the best known results

Bibai, J. , Savéant, P. , Schoenauer, M. and Vidal, V.. On the Benefit of Sub-Optimality within the Divide-and-Evolve Scheme. In P. Merz and P. Cowling, eds., EvoCOP 2010, LNCS 6022, pp 23-34, Springer-Verlag. 2010.

Comparing to state-of-the-art planners

IPC benchmarks domains

- Classical planning
  - 9 domains
  - 256 instances
- Cost planning
  - 8 domains
  - 240 instances
- Temporal planning
  - 9 domains
  - 240 instances
- Total 736 instances

IPC (International Planning Competition) rules

- Limited time : 30 minutes per instance
- Score(Planner, Instance) = $\frac{\text{cost of best known plan}}{\text{cost of generated plan}}$

  in [0,1], 0 if unsolved, 1 if best known quality
- Total score = sum of scores per instances

Discussion

- Score only depends on plan quality
- In a strictly limited runtime
- Stochastic planners: Best (median?) quality out of 11 runs

Results

	Instances solved				Quality score			
Planner	YAHSP	LAMA	LPG	DAEx	YAHSP	LAMA	LPG	DAEx
Classical (256)	218	238	204	241	182.72	229.26	196.56	230.70

Planner	YAHSP	LAMA		DAEx	YAHSP	LAMA		DAEx
Cost (240)	219	221		222	123.58	182.41		184.55

Planner	YAHSP	TFD	LPG	DAEx	YAHSP	TFD	LPG	DAEx
Temporal (240)	217	182	198	219	139.96	148.44	186.46	195.97

DAE: Same parameters for all instances

Bibai, J. , Savéant, P. , Schoenauer, M. and Vidal, V.. An Evolutionary Metaheuristic Based on State Decomposition for Domain-Independent Satisficing Planning. In R. Brafman et al., eds, Proc. ICAPS 2010, pp 15-25, AAAI Press, 2010.

Toward a universal state-of-the-art planner?

Divide and Evolve today:

- Best results on temporal domains
- As good as state-of-the-art on cost and classical domains
- Solves instances that embedded planner cannot solve
- Solutions always close to optimal (at least 90% in quality score)

Room for improvement

- Embedding portfolio of state-of-the-art planners
  - → other types of planning (probabilistic planning on-going)
- Heuristic-based distance in state space
- Improved parameter tuning
- Multi-objective AI planning (no competitor so far)