ARTIFICIAL INTELLIGENCE

Edge Matching Puzzle

Anirudha Sahu Gangaprasad Koturwar

Mentor- Prof. Amit Mukherjee

Indian Institute of technology, Kanpur

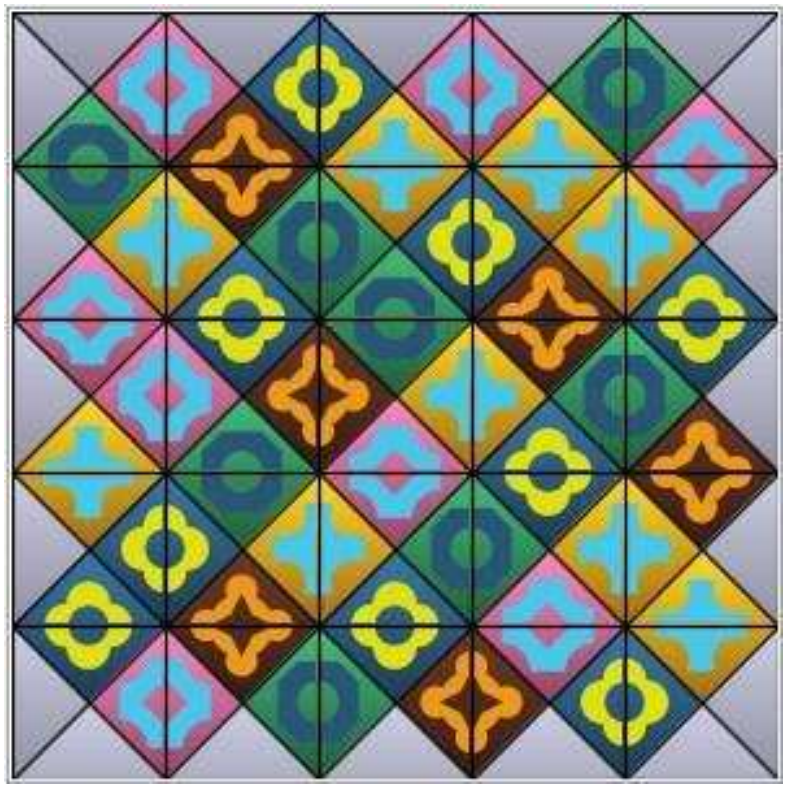
April 4, 2012

The Puzzle and the Challenge

- Published by Christopher Monckton.
- Released on July 2007.
- Offers $ 2M for the first completed solution to the puzzle.
- No complete solution has been provided yet!!!
- Further details at http://www.eternity-puzzle.com/

Puzzle Overview

- Consists of 256 tiles.
- Each tile is colored with 4 different patterns.
- Tiles are to be placed on a 16X16 grid such that the edges of the adjecent tiles should match.



The Difficulties and Challenges

- Huge search space ($256!X4^{256}$ i.e. $1.15X10^{661}$ possibilities.)
- Recursive construction of the solution consumes lot of time.
- Problem of Recursive back-tracking.
- Effective branching factor for A* algorithm is about 381. compared to chess which has only about 100.
- No efficient algorithm can be designed - NP completeness.
- Succesful attempts have matched as much as 396 to 459 edges out of 480.

Previous Works(The Recursive Back-Tracking and GA)

Doc Smiths C++ solver

- Uses recursive Back-Tracking algorithm
  - Looks for suitable tile to be placed.
  - If not found, Removes last placement and repeats.

Joel's JAVA implementation

- Uses same algorithm as Doc smith's solver

Results:(When run on a dual core 2.00Ghz Intel with 1 GB of RAM)

Code	6X6	8X8	10X10
Doc Smith	891msec	91mins	10+hrs
Joel	790msec	107mins	10+hrs

Munoz's Evolutionary Algorithm

- transformed the constraints into optimization objectives and used the optimization power of Genetic Algorithms

(We're mainly following his works and paper)

Previous Works(The Recursive Back-Tracking and GA)

Doc Smiths C++ solver

- Uses recursive Back-Tracking algorithm
  - Looks for suitable tile to be placed.
  - If not found, Removes last placement and repeats.

Joel's JAVA implementation

- Uses same algorithm as Doc smith's solver

Results:(When run on a dual core 2.00Ghz Intel with 1 GB of RAM)

Code	6X6	8X8	10X10
Doc Smith	891msec	91mins	10+hrs
Joel	790msec	107mins	10+hrs

Munoz's Evolutionary Algorithm

- transformed the constraints into optimization objectives and used the optimization power of Genetic Algorithms

(We're mainly following his works and paper)

Previous Works(The Recursive Back-Tracking and GA)

Doc Smiths C++ solver

- Uses recursive Back-Tracking algorithm
  - Looks for suitable tile to be placed.
  - If not found, Removes last placement and repeats.

Joel's JAVA implementation

- Uses same algorithm as Doc smith's solver

  Results:(When run on a dual core 2.00Ghz Intel with 1 GB of RAM)

Code	6X6	8X8	10X10
Doc Smith	891msec	91mins	10+hrs
Joel	790msec	107mins	10+hrs

Munoz's Evolutionary Algorithm

- transformed the constraints into optimization objectives and used the optimization power of Genetic Algorithms

(We're mainly following his works and paper)

Previous Works(The Recursive Back-Tracking and GA)

Doc Smiths C++ solver

- Uses recursive Back-Tracking algorithm
  - Looks for suitable tile to be placed.
  - If not found, Removes last placement and repeats.

Joel's JAVA implementation

- Uses same algorithm as Doc smith's solver

Results:(When run on a dual core 2.00Ghz Intel with 1 GB of RAM)

Code	6X6	8X8	10X10
Doc Smith	891msec	91mins	10+hrs
Joel	790msec	107mins	10+hrs

Munoz's Evolutionary Algorithm

- transformed the constraints into optimization objectives and used the optimization power of Genetic Algorithms

(We're mainly following his works and paper)

The Genetic Algorithm

[1] **Simulation of the process of evolution on Earth.**

Steps Involved:

Initialization

- Creating bunch of "organisms" with unique set of "genes" i.e. generating a "generation".

Simulation:

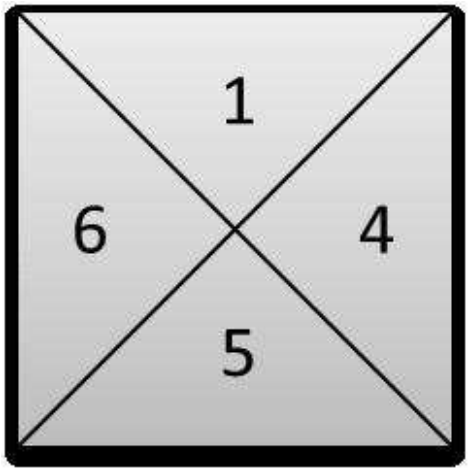
- Every organism is alloted fitness value depending on some criterion.
- Best orgnisms from the generation chosen depending on fitness value.
- Mutation, Crossover, Reproduction.
- Next generation of organisms created.
- Search for the Perfect organism
  -If found Return;
  -if not found Continue;

[1] courtesy: http://www-cs-students.stanford.edu/ jl/Essays/ga.html

Applying Genetic algorithm

Tile Representation:

- Colour of the edge of the tile is replaced with specific numbers notations.



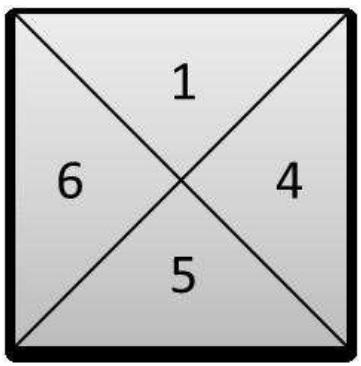
Applying Genetic algorithm

Tile Representation:

Rotations of tiles

Orientation	Rotation(clockwise)
1	original
2	90^o
3	180^o
4	270^o

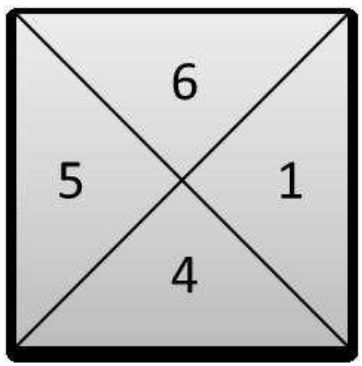
Example:



ori..

1 4 5 6 1

rotation



ori..

6 1 4 5 2

Applying Genetic algorithm

Terminologies:

- Individuals

  Represented as 2D matrix with each entry as tile number and its orientation.

  For ex 113:3 represents tile no 113 with rotation 3.

  - Retains the properties just like genes in chromosomes.
  - Properties like fitness are associated with Individuals.

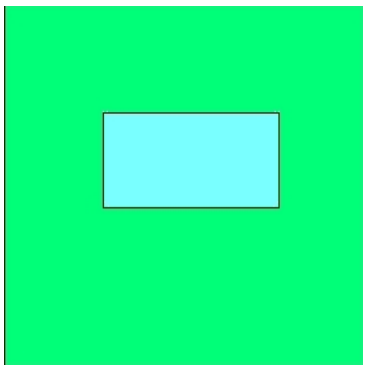
- Generation

  A set of certain number of individuals.

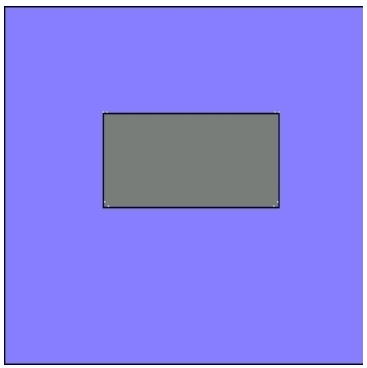
  - Crossover, mutation, Elitism applied on one generation to get next one.

- Crossover

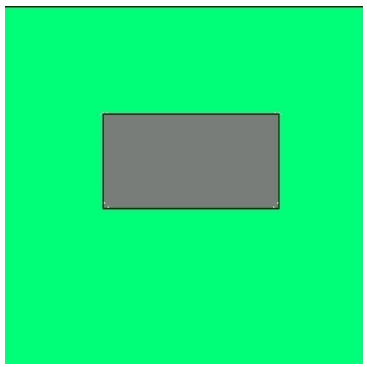
  Copying of the propeties of two parent individuals into the offspring.



Parent 1



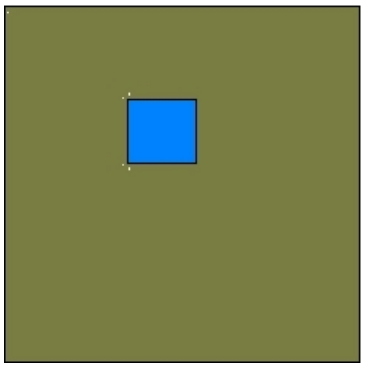
Parent 2

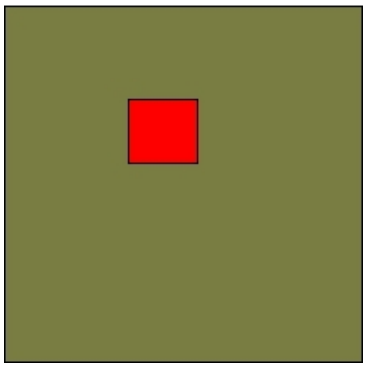


Child

- Mutation

  Propeties of the individual are modified to create better offspring.





  Mutation can be carried out by rotating, swapping etc.

- Elitism

  A certain no of best individuals are selected to retain their properties as it is for next generation.

- Fitness

The property of the individual to be optimized.

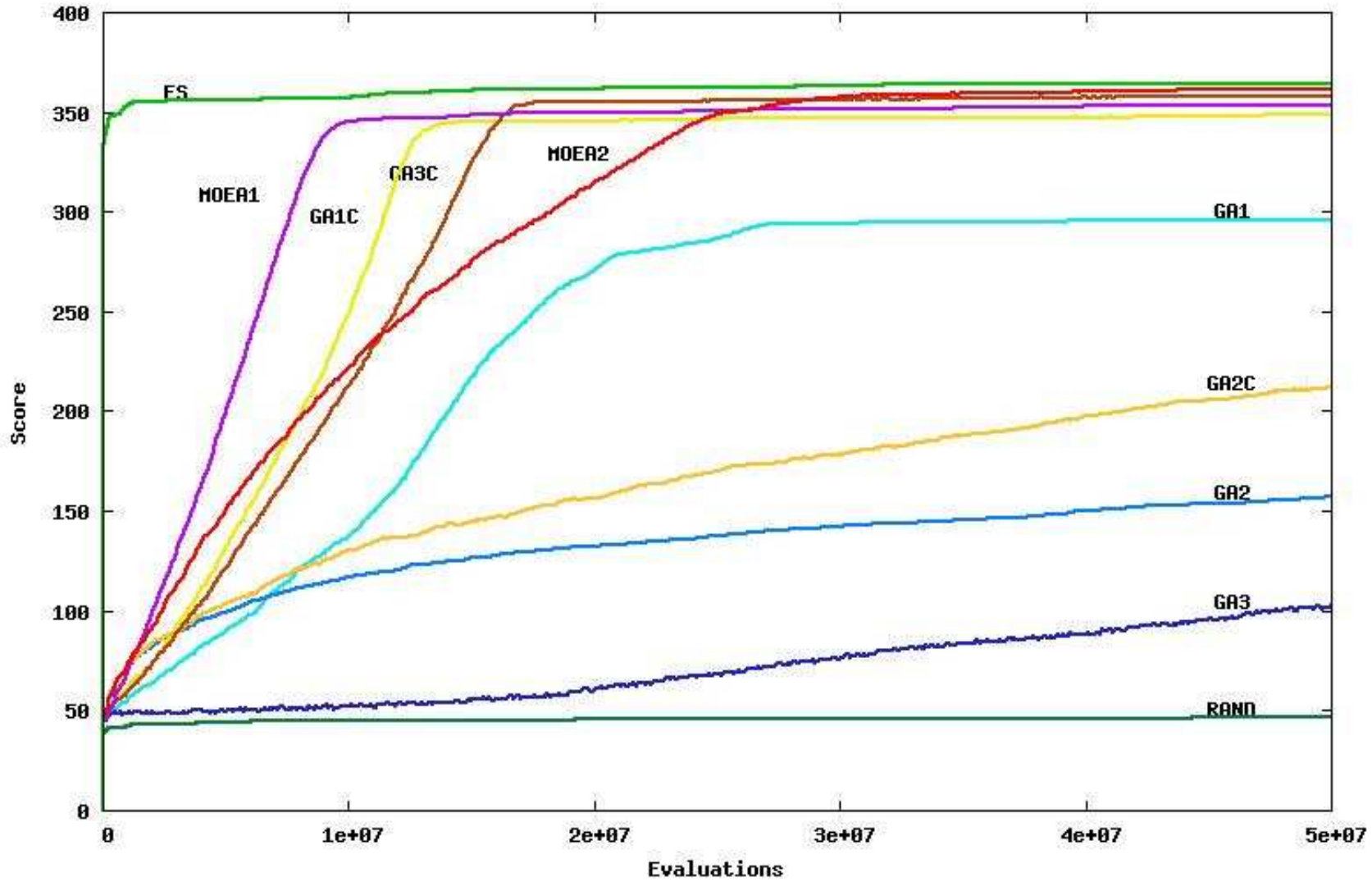
Normal Fitness: $1 - \left(\frac{score}{480}\right)$

in case of multiple objectives:

Combined Fitness: $1 - \left(\frac{1}{k} . \sum_{i=1}^{k} \frac{\text{objective}_i}{\text{Maxobjective}_i}\right)$

where k is number of objectives

Results: Previous Works



[2]courtesy: Paper in "Evolutionary Genetic Algorithms in a Constraint Satisfaction Problem: Puzzle Eternity II"

Problems faced

- Problem of incompleteness - Problem of Local maxima.
- optimization of the Genetic operations.
- Fitness Criteria.
- Selection criteria.

Results

Evaluatons	Elitism	Crossover	Mutation	BestFitness
1000000000	10	179	50	292
100000000	10	179	50	273
10000000	0	1	0	48
10000000	0	0	1	90
10000000	1	0	0	NoCompl
10000000	1	1	0	159
10000000	1	1	1	262
10000000	1	1	2	268
10000000	1	2	1	254
10000000	2	1	1	257
100000	10	14	12	10
10	2	2	10	1
10000	2	2	10	2

References

- Evolutionary Genetic Algorithms in a Constraint Satisfaction Problem: Puzzle Eternity II

  by "Jorge Munoz, German Gutierrez, and Araceli Sanchis", University Carlos III of Madrid Avda. de la Universidad 30, 28911 Legan's, Spain (2009)

- Solving the Eternity II Puzzle Using Evolutionary Computing Techniques

  (A thesis) by "Papa Ousmane Niang", Concordia University, Montreal, Quebec, Canada (2010)

- Codes of Jorge Munoz

  Codes that used for paper "Evolutionary Genetic Algorithms in a Constraint Satisfaction Problem: Puzzle Eternity II"