Introduction To Theory Solver

Vu Nhat Hoang

December 3, 2024

Contents

1 Overview

The system have 2 domains: Game Theory and Stable Matching. Each domain has its many variant, but the system currently support

Game Theory Support only Normal form game

Stable Matching Support many-many matching

2 Game Theory

As mentioned above, the sole game representation is in Normal form where all the player make decision simultaneously or don't have information of others decision , the most notable example is the "Prisoners' Dilemma" problem

This model is emulated by the system using the passed in data:

- Number of players
- Payoff function
- Fitness function
- Special player
- Strategy and their property matrix
- Conflict

After these data is provided to the system, the payoff of each player's strategies are calculated with the payoff function. Note that the same strategy can have different payoff with different value. It can be interpreted as

	strategy 1	strategy 2
player 1	100	45
player 2	78	244

Detail about how payoffs are calculated is explained here

Conflict is an additional functionality, which basically discourages any 2 players to make specific strategies at any iteration

2.1 Flow of Event

With each iteration, all the player simultaneously make a decision, which have their corresponding payoff, then the chosen Genetic Algorithm will use the fitness function to evaluate the payoffs list
Here is how fitness is calculated

2.2 Payoff

The strategies available to any player can be represented as a table as follow

	property 1	property 2
strategy A	188	1.2
strategy B	19.7	129

Payoff is calculated using the payoff function, which default to the sum of a strategy's properties or a custom function using the syntax `p<column index> [<arithmatic> p<column index>]` with index starting from 1

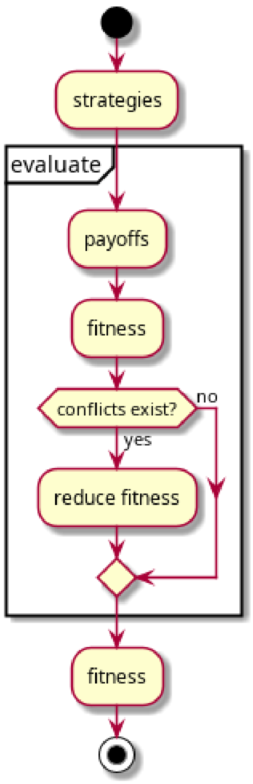

Figure 1: Solution evaluation

3 Stable Matching

The many-many variation differs from the original problem define by Gale and Sharply by introducing individuals' capacity. Therefore, the model is represented in the system as follow:

- Capacities
- Properties
  - Value
  - Weight
  - Requirement
- Set indices
- Exclude pairs
- Evaluate functions
- Fitness function

Evaluate function is specific to individuals within a set which is used to compute the preference of one individuals to others based on the components of individual's properties detail of the calculation is here

Exclude pair is an additional functionality which stops 2 specific individual to be matched

Capacity is the maximum number of matches an individual may have

Fitness function calculate the final score from the list of preference from the matching result, this default to be the sum of the list

Set index a list of integer indicating an individual's set. Example: [1, 1, 2] the 1st and 2nd individuals belong to Set 1, and the 3rd individual belonged to the Set 2

3.1 Flow of Event

With each iteration, a queue of individuals is used to generate stable matching which is then used to calculate fitness. The order of the queue will be adjusted by the Genetic Algorithm

Here is a visualization:

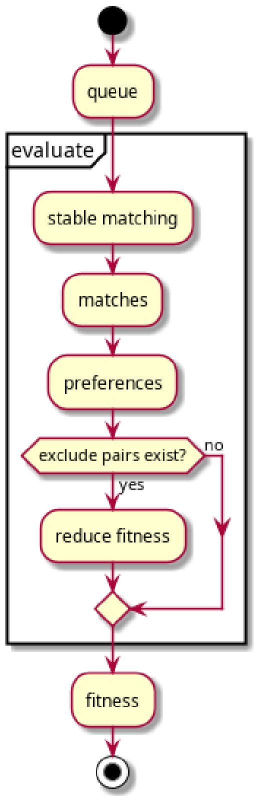

Figure 2: Solution evaluation

3.1.1 Stable Matching

As mentioned the system expand on the original problem by introducing many-many matching. This require some modification to the Gale-Shapley Algorithm

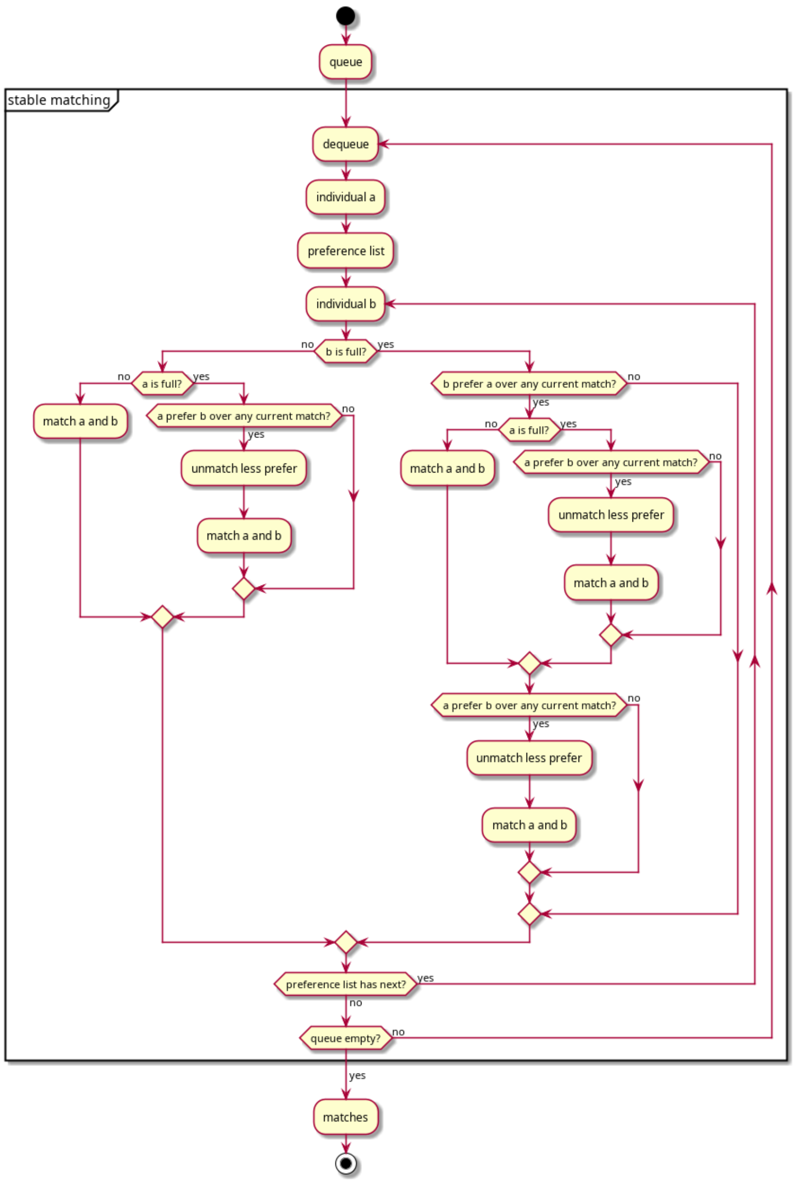

Figure 3: Stable Matching

3.2 Preference

The preference of individual A to individual B based on 3 factors

- Requirement
- Value
- Weight

These factors are presented in every property of an individual and is used by the Evaluate function to calculate preference

Example

Individual A	Weight	Value	Requirement
Property 1	12	12	12–
Property 2	13	13	13
Property 3	23	18	18:100

Individual B	Weight	Value	Requirement
Property 1	12	12	12–
Property 2	13	13	13
Property 3	23	18	18:100

Common annotation R_i, W_i, P_i

- i represent Property row
- R represent Requirement column
- P represent Value column
- W represent Weight column

Example: R_1 means Requirement of the Property 1

3.2.1 Custom Evaluation

A custom evaluate function is $R_2 + W_1 * P_1$ The value of R_2 and W_1 will be from A and P_1 will be from B

W_1 and P_1 are very straight forward, While R_2 has more nuance. with 3 syntax give different result

- $x--$ return x
- $x : y$ return $\frac{x+y}{2}$
- $x++$ return x
- x return x

Apply these rules, the preference of A to B should be $13 + 12 * 12 = 157$

3.2.2 Default Evaluation

Default evaluation will use all property to calculate preference. The formulation can be expressed as

$$P_A(B) = \sum_{i=1}^{N} R_i(P_i) * W_i$$

- N the number of property of an individual
- R_i the requirement function for property i of A
- W_i the weight for property i of A
- P_i the value for property i of B

Default evaluation behaves a little different with Requirement, which is used as a function to transform the Value factor

$$R_i(P_i) = x--(P_i) = \begin{cases} 2 & if\ x = 0 \\ 0 & if\ P_i > x \\ \frac{x+|P_i-x|}{x} & else \end{cases}$$

$$R_i(P_i) = x++(P_i) = \begin{cases} 2 & if\ x = 0 \\ 0 & if\ P_i < x \\ \frac{x+|P_i-x|}{x} & else \end{cases}$$

$$R_i(P_i) = x:y(P_i) = \begin{cases} \frac{\frac{|y-x|}{2} - |\frac{x+y}{2}| + P_i}{\frac{|y-x|}{2}} + 1 & if\ P_i \in [x,y] \\ 0 & else \end{cases}$$

$$R_i(P_i) = x(P_i) = \begin{cases} 0 & if\ P_i < 0\ or\ P_i > 10 \\ 0 & if\ |P_i - x| > 7 \\ 1 & if\ |P_i - x| > 5 \\ \frac{10-|P_i-x|}{11} & else \end{cases}$$

Apply these rules, A preference to B should be

$$\begin{aligned} P_A(B) &= R_1(P_1) * W_1 + R_2(P_2) * W_2 + R_3(P_3) * W_3 \\ &= 12--(12) * 12 + 13(13) * 13 + 18:100(18) * 23 \\ &= \frac{12+|12-12|}{12} * 12 + 0 * 13 + (\frac{\frac{|100-18|}{2} - \frac{100+18}{2} + 18}{\frac{|100-18|}{2}} + 1) * 23 \\ &= 12 + 0 + 23 \\ &= 35 \end{aligned}$$