Jupiter User Guide

作成日　平成 29 年 12 月 27 日

氏名　福井 智哉

はじめに

Jupiter は自動交渉シミュレーション環境である．

自動交渉という研究分野が存在し，またそのシミュレーション環境として Genius[1] が挙げられる．しかし，Genius はその設計思想や開発時期の点から，昨今研究がさかんである機械学習技術の応用が難しい．そこで，Jupiter という自動交渉シミュレーション環境を構築した．Jupiter では自動交渉のプロトコルとして Stacked Alternating Offers Protocol(SAOP)[2] に基づくエージェントによる自動交渉のシミュレーション環境を提供する．Jupiter の特徴の 1 つとして，先行研究である Genius で開発されたエージェントや交渉ドメインファイルを，用いることを可能としている．

本ユーザーガイドでは，Jupiter の簡単な使い方を説明していく．

1 章では，Jupiter で取り扱う自動交渉のプロトコルや交渉ドメインの設定について述べる．2 章では，Jupiter を実行するための環境と，その実行の仕方について述べる．3 章では，Jupiter の全体の構成と Jupiter の API について説明する．

目 次

図 目 次

第1章

Jupiterで取り扱う交渉プロトコルと交渉ドメイン

1.1 序言

本章では，まず初めに Jupiter で取り扱う交渉のプロトコルについて述べる．交渉プロトコルでは，交渉の進め方について述べている．次に Jupiter で取り扱っている交渉ドメインについて述べる．交渉ドメインには，交渉結果における，各交渉参加エージェントの獲得する効用値について述べられている．

1.2 交渉プロトコル

Jupiterでは交渉プロトコルとして，Stacked Alternating Offers Protocol(SAOP)[2]を採用している．SAOPでは，交渉参加エージェントが順番に，合意候補案の中から1つ選び提案しあうことにより，交渉を進行させる．交渉時間の種類にはターン制と時間制の2種類が存在し，各エージェントの行動にはOffer，Accept，Endnegotiationの3種類存在する．

1.2.1 交渉時間の種類

1. ターン制

   全参加エージェントの合意またはEndNegotiationが発生したとき，もしくはあらかじめ決められたターンまで合意が得られかったときまで交渉を続ける．例えば180ターンで参加エージェント数が2の交渉設定の場合，交渉過程で合意またはEndNegotiationが発生したとき，もしくは2つのエージェントがそれぞれ180回行動を行なったときに交渉が終了する．

2. 時間制

   現実時間が交渉の制限時間となる．例えば180秒で参加エージェント数が2の場合，交渉過程で合意またはEndNegotiationが発生したとき，もしくは2つのエージェントの行動回数に関係なく現実時間で180秒が経過したときに，交渉が終了する．

1.2.2 各エージェントの行動

エージェントの行動には，Offer，Accept，EndNegotiation の 3 種類が存在する．また交渉参加エージェントは，交渉参加エージェントの順番に関係なく，他エージェントの行動を全て認知することができる．しかし，行動を起こすには，自分の手番が回ってくるまで待たなければならない．

1. Offer

   合意案候補から 1 つ選び，提案する．

   他エージェントからの Offer を通知されたあとに，自エージェントが Offer を行なった場合，その他のエージェントの Offer は自動的に却下されたことになる．

2. Accept

   最近の他エージェントからの Offer を受け入れることを意味する．

   あるエージェントの Offer を，そのエージェント以外の全エージェントが連続で Accept した場合，その交渉は合意に至ったとして判定され，交渉が終了する．

3. EndNegotiation

   交渉の強制的な終了を意味する．他のエージェントの行動に関わらず，交渉を終了することができる．

   他エージェントが交渉において，割引効用が大きく，まったく妥協しない場合に有用な選択肢となる．割引効用に関しては，交渉ドメ

インの節で述べる．

1.3 交渉ドメイン

この section では，交渉ドメインについて述べる．交渉ドメインには，交渉結果において，各エージェントが取得する効用値に関する情報が記されている．Jupiter では，効用空間は線形効用空間をサポートしており，交渉ドメインは Genius で用いられている xml 形式のファイルを読み出す仕様になっている．現在の Jupiter では，交渉ドメインを作成する機能を提供していないため，注意されたい．

交渉ドメインに記述されている内容を説明する前に，合意案候補 (bid) を数式で示す．$\boldsymbol{s}$ を合意案候補，n を論点の数，I_i を i 番目の論点で取りうる選択肢の集合と定義すると，各合意案候補 (bid) は以下の式により定義される．

$$\boldsymbol{s} = \{(s_1, s_2, \ldots, s_n) | s_i \in I_i, i = 1, 2, \ldots, n\}$$

以上をふまえて，交渉ドメインに記述されている以下の 3 つの内容を説明する．

1. 各論点の重みと，各論点における各選択肢の効用値

   各合意案候補の効用値 $U(\boldsymbol{s_k}; \boldsymbol{w})$ は以下の式で表される．

$$U(\boldsymbol{s}; \boldsymbol{w}) \quad = \quad \sum_{i=1}^{n} w_i \cdot eval(s_i)$$

$$\sum_{i=1}^{n} w_i = 1$$
$$eval(s_i) = \frac{value(s_i)}{\arg\max_{s_i \in I_i} value(s_i)}$$

w_i は i 番目の論点における重みを表し，$0 \leq w_i \leq 1 (1 \leq i \leq n)$ である．$value(s_i)$ は I_i において s_i を選んだ場合の効用値を表す．$w_i (1 \leq i \leq n)$ と $value(s_i) (s_i \in I_i)$ の値は交渉ドメインに記されている．補足として，定義から $0 \leq eval(s_i) \leq 1$ のため，$0 \leq U(\boldsymbol{s}; \boldsymbol{w}) \leq 1 (\forall \boldsymbol{s})$ である．

2. 留保価格

   EndNegotiation や合意が得られずに交渉が終了した際に得られる効用値を表す．留保価格を r，合意失敗を e とすると，以下の式で定義される．

$$U(e) = r$$

3. 割引効用

   交渉が終わった際に，交渉にかかった時間に応じて，各エージェントが得られる効用値を割り引くために使われる値を表す．各エージェントが得られる割引済みの効用値を U_d，割引効用を d，正規化された交渉の経過時間を $t (0 \leq t \leq 1)$ とすると

$$U_d(\boldsymbol{s}, t; d) = U(\boldsymbol{s}) \cdot d^t$$

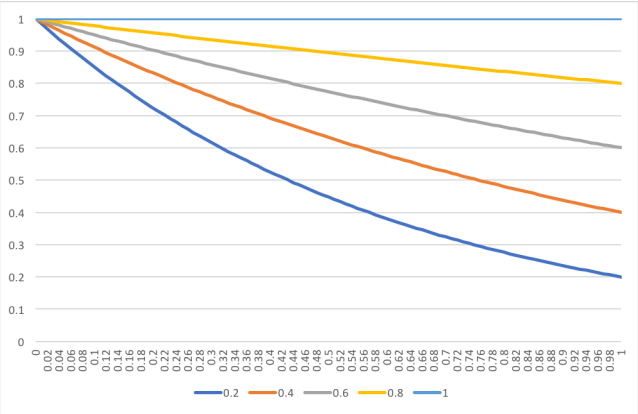

図 1.1: 割引効用込みの獲得効用

$$U_d(e, t; d) = U(e) \cdot d^t$$

として表される．割引効用が $0.2, 0.4, \ldots, 1.0$ のときの d^t を図 1.1 に示す．

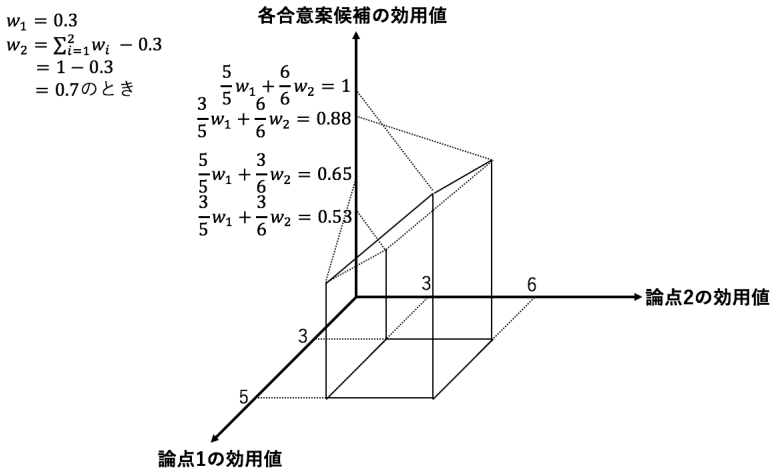

図 1.2: 論点数 2，各論点の選択肢が 2 の場合の効用空間

1.3.1　効用値の計算例

ここまでで交渉ドメインの説明をしてきたが，ここで実際に獲得効用 $U(\boldsymbol{s})$ の計算例を示す．今回計算する交渉ドメインは図 1.2 で表される．数式で表すと $n = 2$，$|I_1| = 2$，$|I_2| = 2$，$w_1 = 0.3$，$w_2 = 0.7$ であり，各論点におけるそれぞれの選択肢の効用値が以下である．

$$value(s_{11}) = 3$$

$$value(s_{12}) = 5$$

$$value(s_{21}) = 3$$

$$value(s_{22}) = 6$$

合意案候補全体の集合を $\boldsymbol{S}$ とすると，

$$\boldsymbol{S} = \{(s_{11}, s_{21}), (s_{11}, s_{22}), (s_{12}, s_{21}), (s_{12}, s_{22})\}$$

である．(s_{11}, s_{21}) について，獲得効用の計算例を示す．

$$\begin{aligned} U((s_{11}, s_{21}); \boldsymbol{w}) &= w_1 \cdot \frac{value(s_{11})}{\arg\max_{s_1 \in I_1} value(s_1)} + w_2 \cdot \frac{value(s_{21})}{\arg\max_{s_2 \in I_2} value(s_2)} \\ &= 0.3 \cdot \frac{3}{5} + 0.7 \cdot \frac{3}{6} \\ &= 0.53 \end{aligned}$$

他の合意案候補についても同様にして

$$\begin{aligned} U((s_{11}, s_{22}); \boldsymbol{w}) &= 0.88 \\ U((s_{12}, s_{21}); \boldsymbol{w}) &= 0.65 \\ U((s_{12}, s_{22}); \boldsymbol{w}) &= 1 \end{aligned}$$

となる．

第2章

Jupiterの起動方法

2.1　序言

本章では，Jupiter の起動方法について示す.

2.2　動作環境

Jupiter は Python3 で書かれており，以下の環境において動作する.

- python がバージョン 3.6 以上であること，

- 「numpy」,「pandas」,「py4j」,「matplotlib」モジュールがそれぞれインストールされていること.

参考として筆者の動作環境を以下に示す.

- OS
  - macOS Sierra 10.12.5

- Python のバージョン
  - 3.6.1

- Python3 パッケージ一覧
  - cycler==0.10.0
  - matplotlib==2.1.1
  - numpy==1.13.3
  - pandas==0.21.1
  - py4j==0.10.6
  - pyparsing==2.2.0
  - python-dateutil==2.6.1
  - pytz==2017.3
  - six==1.11.0

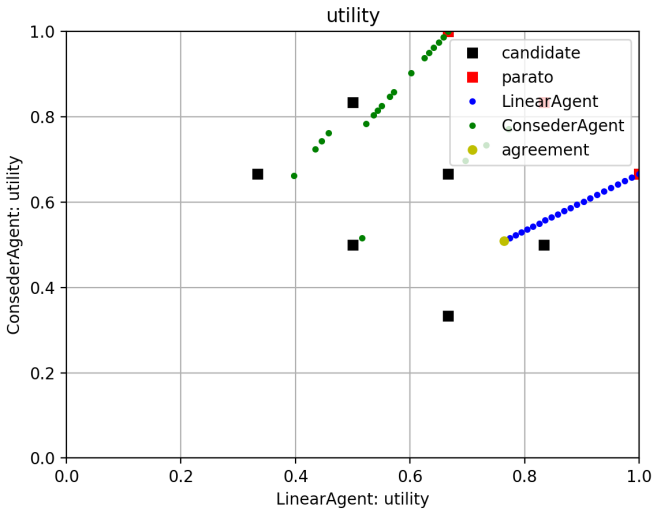

図 2.1: Jupiter の実行例

2.3　Jupiter を実行する

この節では Jupiter のプログラムを実行するまでを解説する．

1. まずはじめに下記のサイトより，プロジェクトをダウンロードする．

   `https://github.com/TomoyaFukui/Jupiter`

2. src/jupiter/jupiter.py を実行する．

   実行できた場合，二者間交渉が始まる．

以上の手順により，jupiter を実行することができる．図 2.1 のような画像が表示されれば，実行成功である．

2.4 Jupiter の実行方法

2.4.1 Jupiter の実行ファイルを作成する

Listing 2.1: src/main.py

```
1  from jupiter.jupiter import Jupiter
2  from jupiter.negotiationRule import TypeOfNegotiation
3
4  if __name__ == '__main__':
5    jupiter = Jupiter(TypeOfNegotiation.Turn, 180, 'Atlas3/triangularFight
       .xml',
6      'Atlas3/triangularFight_util1.xml', 'Atlas3/triangularFight_util2.
         xml')
7      jupiter.set_agent('LinearAgent')
8      jupiter.set_agent('ConcederAgent')
9      jupiter.do_negotiation(is_printing=True, print_times=1)
10     jupiter.display.show()
```

この章では Jupiter の，より詳しい実行方法について述べる．以下の手順により Listing2.1 をまず実行して欲しい．

1. main.py を作成し，src フォルダ直下へ保存する．

2. main.py の中身を Listing2.1 に書き換える．

3. main.py を実行する．

   実行できた場合，二者間交渉が始まる．

以下，main.py の解説に加え，Jupiter を実行する際の流れを説明する．

1. jupiter モジュールから Jupiter クラス，negotiationRule モジュールから TypeOfNegotiation クラスをそれぞれ import する．

2. Jupiter クラスを作成する．コンストラクタに指定する引数は以下．

   1 つ目の引数として，交渉の種類を指定する．ターン制と時間制がある．

   2 つ目の引数として，交渉の長さを指定する．時間の場合，単位は秒となる．

   3 つ目の引数として，交渉の設定ファイルを指定する．

   4 つ目以降の引数として，効用情報の書かれたファイルを指定する．

   Jupiter の仕様として，4 つ目以降のファイルの数が，同時交渉エージェントの数となる．

3. 交渉に参加するエージェントを指定する．

   Listing2.1 では，LinearAgent と ConcederAgent を指定している．

   引数は str 型にして，クラスの名前と同じ変数を渡す．

Jupiter を実行する際は，上記の流れを行うことにより，交渉シミュレーションを行うことができる．より詳しい機能については，jupiter.py やリファレンスを参照されたい．

2.4.2 ドメインファイルを追加する

Jupiter では現在，Genius で作成されたドメインファイルを扱えるようになっており，ドメインファイルの作成機能に関しては，サポートしていない．Genius で作成したファイルを追加したい場合は「src/domain」ディレクトリに保存することで呼び出せる．

2.4.3 エージェントを作成する

本章では，エージェントを作成し，Jupiter で動かすところまでを説明する．

1. myAgent.py を作成し,「src/agents」フォルダへ保存する．
2. myAgent.py の中身を Listing2.2 に書き換える．
   各関数についてのより詳しい説明に関しては，AbstractAgent クラスのリファレンスを参照されたい．
3. jupiter.py 中の「from concederAgent import*」の次の行に「from myAgent import *」を追加する．

上記の手順を踏むことにより，エージェントを追加することができる．

Listing 2.2: src/agents/myAgent.py

```
1 import sys
```

```
2  import os
3  sys.path.append(os.path.dirname(os.path.abspath(__file__)) + '..')
4  import abstractAgent
5  import agentAction
6  import abstractUtilitySpace
7  import negotiationRule
8
9  class MyAgent(abstractAgent.AbstractAgent):
10     def __init__(self, utility_space: abstractUtilitySpace.
         AbstractUtilitySpace,
11                  negotiation_rule: negotiationRule.NegotiationRule,
                      agent_id: int, agent_num:int):
12         self.__utility_space = utility_space
13         self.__rule = negotiation_rule
14         self.__agent_id = agent_id
15         self.__opponent_bid = None
16
17     def receive_action(self, agentAction_: agentAction.AbstractAction):
18         if isinstance(agentAction_, agentAction.Offer):
19             self.__opponent_bid = agentAction_.get_bid()
20
21     def send_action(self):
22         def get_conssetion_value():
23             return (1.0 - self.__rule.get_time_now())
24
25         if self.__opponent_bid is not None and \
26             get_conssetion_value() < self.__utility_space.get_utility(
                  self.__opponent_bid):
27             return agentAction.Accept(self.__agent_id)
28
29         bid_offer = self.__utility_space.get_bid_above_concession_value(
              get_conssetion_value())
30         return agentAction.Offer(self.__agent_id, bid_offer)
31
32     def receive_start_negotiation(self):
33         self.__opponent_bid = None
```

```
34
35      def get_name(self):
36          return 'MyAgent'
```

第3章

終わりに

以上でUser Guideを終わりとする．より詳しい説明や機能に関しては，リファレンスを参照されたい．また実際にJupiterを使っていただければ幸いである．

参考文献

[1] Lin, Raz, et al.: "Genius: An integrated environment for supporting the design of generic automated negotiators." Computational Intelligence 30.1 (2014): 48-70.

[2] Aydoan, Reyhan, et al.: "Alternating offers protocols for multilateral negotiation." Modern Approaches to Agent-based Complex Automated Negotiation. Springer International Publishing, 2017. 153-167.