3G2-01

Prologによるゲームプレイヤーのモデリング

Modeling game players by using Prolog

犬童健良[*1]
Kenryo Indo

[*1] 関東学園大学経済学部
Faculty of Economics, Kanto Gakuen University

This paper describes some modeling for rational or bounded rational game players in accordance with game theory by using Prolog, a logic programming language. The reasoning about mutual knowledge of players is also modeled.

1. ゲームプレイヤーのインテリジェンス

社会では自分の選んだ行動や考え方が合理的であるか否かは，少なくとも部分的には他者の行動選択に依存しており，またそれは相手にとっても同じである．ゲーム理論はそのような社会状況をゲームとして定式化し，ゲームの参加者であるプレイヤーたちが，いかに合理的に行動するのかを説明する．

ゲーム理論は経済学を中心に諸分野で利用されている[例えば Muto 01, Imai 02]．ボードゲームで勝つための具体的なアルゴリズムを研究している AI とは対照的に，基本的には，ゲームプレイヤーは十分知的である，すなわち最適化問題としてゲームを解く能力があると仮定し，その理論的帰結を吟味する．しかし近年のゲーム理論研究は，たんなる行動レベルの最適化から，プレイヤーの内部構造へとその分析を深めている．例えば以下にあげるような，さまざまな切り口で発展 改良がなされており，AI ないしエージェントシステム研究との接点も少なからずある．

- 相互知識や共通知識の定式化 その緩和
- メカニズムデザイン 遂行理論 分権的システムの分析．
- 交渉プロトコールや入札のデザイン．Eコマースへ応用
- チューリングマシンやオートマトンによる計算論的研究
- その他の限界合理性モデル．進化ゲーム，学習，曖昧信念，非期待効用，情報処理エラーや不完全想起の場合．
- 実験経済学的手法 ゲームデザインへの積極的応用

以下では筆者が行った Prolog を用いたゲーム分析のモデリング事例をいくつか紹介する．第 2 節では古典的なゲーム分析である標準形と提携形およびメカニズムデザイン理論のモデリングを説明する．第 3 節と第 4 節で展開形ゲームと共通知識をそれぞれモデリングする．最後に第 5 節でまとめを述べる．

2. 行動水準でのゲームモデリング

標準形ゲームでは最適応答がプレイヤーの合理性の特徴だが，提携形ゲームでは個人合理性，全体合理性，非支配(非ブロック)関係などの制約条件のセットで表現される．しかし，その分析は行動的水準にとどまっており，プレイヤーの心的モデルや心的プロセスには直接かかわらない．メカニズムデザイン理論はその延長にあるが，エージェントの社会で分権的に実行できることとできないこととを峻別するための理論として注目される．

2.1 標準形ゲームとナッシュ均衡

標準形ゲームでは，各プレイヤーの可能な行動の組に対して利得ベクトルを定義する．ナッシュ戦略とは最適応答であり，またその均衡点はその行動組自身に対して各プレイヤーが利得最大化行動をしていること，またはそのときのゲームの結果である．ちなみに，ゲームプレイヤーの最大化行動は期待効用理論によって基礎付けされる．

例として，2人標準形ゲーム s1 の利得表を図1に示す．これは各プレイヤーがそれぞれ2つの純粋戦略を持つ同時手番のゲームである．図1中の＊印は最適応答である．このゲームのナッシュ均衡は[1,1]と[1,0]の2つである．いずれも，プレイヤーが単独で自分自身の行動を変更しようとする動機がないことが確かめられるという意味で，自己拘束的な選択をしている．

Payoff of game s1		Act of player 2 z	w
Act of player 1	x	[1, 1] * *	[-2, 0] *
	y	[1, 0] * *	[-1, -1] *

図 1. 標準形ゲームの利得表

一方，唯一の均衡点が，誰にとっても別の非均衡点よりも明らかに悪い囚人ジレンマゲームのような例もある．実際，図 1 の利得表の非対角部分を変更したゲームs2では，[-1,-1]が唯一の均衡となる．Prolog によるこれらの例題のモデリングと純粋ナッシュ戦略での均衡を求めるプログラムを以下に示す．

- 標準形ゲームの Prolog モデル

```
game(s1, acts([x,z]), payoffs([1,1]) ).
game(s1, acts([x,w]), payoffs([-2,0]) ).
game(s1, acts([y,z]), payoffs([1,0]) ).
game(s1, acts([y,w]), payoffs([-1,-1]) ).
game(s2, acts([x,z]), payoffs([1,1]) ).
game(s2, acts([x,w]), payoffs([-2,2]) ).
game(s2, acts([y,z]), payoffs([2,-2]) ).
game(s2, acts([y,w]), payoffs([-1,-1]) ).
```

- ナッシュ均衡を求めるプログラム

```
nash(G,[S1,S2],[P1,P2]):-
   game(G,acts([S1,S2]),payoffs([P1,P2])),
   \+ (game(G,acts([_,S2]),payoffs([Px,_])),Px>P1),
   \+ (game(G,acts([S1,_]),payoffs([_,Py])),Py>P2).
```

- 実行例

```
?- nash(G,Acts,Payoffs).

G = s1
```

連絡先：関東学園大学経済学部，〒373-8515 群馬県太田市藤阿久 200, Tel: 0276-32-7812, kindo@kanto-gakuen.ac.jp

```
Acts = [x, z]
Payoffs = [1, 1] ;

G = s1
Acts = [y, z]
Payoffs = [1, 0] ;

G = s2
Acts = [y, w]
Payoffs = [-1, -1] ;

No
```

なお詳しくは略すが，容易に N 人へ一般化できる．また戦略の確率化すなわち混合戦略での均衡点の存在はナッシュにより証明されている．モデリングについては省略する．

2.2 提携形ゲームとコア

提携形ゲーム，いわゆる協力ゲームは，プレイヤーの可能な提携(coalition)が生み出す共同利益を特徴関数として定義し，コア，仁，シャプレー値といった多彩な解概念を用い公平な分配案を探求する．一般均衡理論や社会選択理論でとくに重宝された．以下に 3 人提携形ゲームのモデリング事例を示す．

- 提携形ゲーム**（特徴関数）**の Prolog モデル

```
game(c1, coalition([ ]),common_value(0) ).
game(c1, coalition([a]), common_value(0) ).
game(c1, coalition([b]), common_value(0) ).
game(c1, coalition([c]), common_value(0) ).
game(c1, coalition([a,b]), common_value(1) ).
game(c1, coalition([b,c]), common_value(1) ).
game(c1, coalition([a,c]), common_value(1) ).
game(c1, coalition([a,b,c]), common_value(1) ).
```

ちなみにゲーム c1 は多数決投票を表す．プレイヤー集合は N＝{a,b,c}であり，よって可能な提携 S⊆N は 8 通りある．配分(imputation)は，全体合理性：**v（{1,2,3}）**＝x1+x2+x3，および個人合理性:x1, x2, x3≧**v（{i}）**＝0の 2 条件を満たす共同利益分配案(x1,x2,x3)の集合である．コア(core)は，どんな提携 S の下でも互いに支配されない配分の集合であり，したがって誰も明らかな不満(excess)を持たない．

- 配分

```
imputation(game(G),payoff(A)):-
  game(G,form(coalitional),players(N)),
  game(G,coalition(N),value(V)),
  length(N,LN),
  allocation(LN,V,A),
  \+ (
    nth1(K,N,J),
    nth1(K,A,AJ),
    game(G,coalition([J]),value(RJ)),
    RJ > AJ
  ).
```

- コア

```
core(game(G),payoff(A)):-
  game(G,form(coalitional),players(N)),
  imputation(game(G),payoff(A)),
  \+ (
    game(G,coalition(S),value(RY)),
    S \= N,
    selected_sum(S/N,_B/A,AY),
    RY > AY
  ).
```

ちなみに 3 人優加法的提携形ゲームの非空コア存在の必要十分条件は

2・v（{1,2,3}）≧v（{1,2}）＋v（{2,3}）＋v（{1,3}）

であることが知られている．ゲーム c1 は明らかに優加法的だから s1 のコアは空である．また全員提携の値を 3/2 以上にすればコアが生じる．上記 Prolog プログラムを用いてこれを確かめよう．ただし，配分案の整数値近似 (allocation/3)と提携中の配分合計 (selected_sum/3)の各プログラムを別途作成した．

- 実行例（全員提携[a,b,c]の値が1～1.99 のとき）

```
?- core(game(c1),C).

No
```

- 実行例（全員提携[a,b,c]の値が2のとき*）

```
?- core(A,B).

A = game(c1)
B = payoff([1, 1, 0]) ;

A = game(c1)
B = payoff([1, 0, 1]) ;

A = game(c1)
B = payoff([0, 1, 1]) ;

No
```

（*注）利得スケールを 100 倍すれば 150 で初めて[50,50,50]が唯一のコア配分として出力される．

2.3 社会的選択とメカニズムデザイン

前出のゲーム分析では利得が数値表現されていたが，選好順序のみに基づく抽象的なゲームモデルもある．社会的選択(social choice)の環境は，社会の成員の可能な選好順序の集合，社会全体で決めるべき代替案の集合，および各選好組に対して定義された望ましい代替案集合である社会選択規則（SCR）から成り立つ．例えば「どの選好状態でもエージェント 1 とエージェント 2 の好みは一致しないが，3つの代替案のうち c が最悪であることはつねに合意する」といった事実や社会選択規則を，以下のように表せる．

- 選好順序

```
preference(agent(1),state(1),[a,b,c]).
preference(agent(1),state(2),[b,a,c]).
preference(agent(2),state(1),[a,b,c]).
preference(agent(2),state(2),[b,a,c]).
```

- 社会選択規則

```
scc(rule(f),state(1),[a]).
scc(rule(f),state(2),[b]).
```

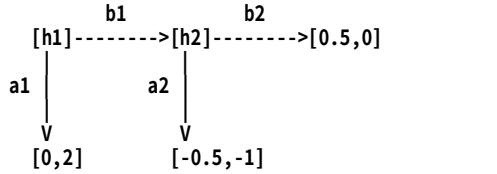
こうして定義された社会選択環境に対して，ゲームを通じて SCR をエージェントたちに分権的に遂行させる問題は，メカニズムデザインないし遂行理論(implementation theory)と呼ばれる．入札や投票へのより具体的な応用も知られるが，より一般的には，SCR を遂行するというのは均衡集合と SCR を一致させるこ

とである．その必要十分条件や遂行用のメカニズムの一般形が研究された．

なお上記の f は明らかに独裁的 SCR だが，筆者の開発したプログラム(impl12.pl) [Indo 02]で確かめると，f は上の選好に対し，それ以外に Maskin 単調性，制限的拒否権なし，弱パレート最適性などを満たすことが分かる．また f は Moore-Repullo の条件 μ2 を満し，ゆえにナッシュ遂行するゲームを実際に作成できるが，紙幅の都合で省略する．

3. ゲーム木における情報モデリング

展開形ゲームでは，プレイヤーの手番の順序のあるゲームが木を使って分析される．ゲーム木の各枝は，各プレイで選択された行動に対応するように表現される．ゲーム木に沿って合理的なプレイを推論し，不完全な合理性しかない均衡を除去する部分ゲーム完全均衡について説明する．



	a1->a2; b1->a2	a1->b2; b1->a2	a1->a2; b1->b2	a1->b2; b1->b2
a1	[0, 2] * *	[0, 2] * *	[0, 2] *	[0, 2] *
b1	[-0.5,-1]	[-0.5, -1]	[0.5, 0] * *	[0.5, 0] * *

図 2 展開形のゲーム木とその行動戦略の利得表

3.1 部分ゲーム完全均衡

ゲーム木のノード集合は，プレイヤーの識別できる情報構造に分割される．完全情報ゲームとはすべての情報集合が単一要素集合，つまりプレイヤーがいつでも自分の手番のノードを識別できる場合である．そうでない場合は不完全情報ゲームと呼ばれる．ただし不完全情報でも通常，一度通った経路は覚えているという完全想起が仮定される．

部分ゲーム完全均衡(subgame perfect equilibrium; SPE)は R. Selten によって導入された．SPE はどの部分ゲームでも最適反応である行動戦略である．ただし部分ゲームとは，各情報集合以降のプレイが，他の部分ゲームと交差せず，一つのゲームの木になっているものとする．例えば，図 2 の上のゲーム木で表される 2 人完全情報ゲームの例 g50(weak) (チェーンストアゲーム) には，じつは 3 つの情報集合と 2 つの部分ゲームがある．

両プレイヤーはともに 2 種類の行動から 1 つを選ぶが，後手は先手の行動を観察できるので，行動戦略は 4 種類ある．これを標準形に変換すると図 2 の下の利得表となり，4つの均衡を得る．実際にプレイされるのは [a1,a2]あるいは[b1,b2]のうちいずれかの行動ペアであり，行動ルールの一部は実際に使われない反事実的条件文(counterfactuals)となる．

3.2 後向き帰納推論

いわゆる後向き帰納推論(backward induction)を図 2 の木に対して適用すると，以下のようになる．後手は情報集合[h2]以降の部分ゲーム，つまり先手のプレイ b1 を観察後，b2 だけが最適反応である．また先手が a1 をプレイすれば，後手はどう選んでも最適反応である．先手はこれを知っているから，b1 が最適であると結論する．こうして[b1,b2]のみが合理的な解として残るが，すべての葉から上のような後向き推論を施せば，部分ゲーム完全均衡に一致することは明らかであろう．

以下は筆者のプログラム(nash1c.pl)を用いて，NE や SPE を求めた結果である．ここでは木生成などのプログラムを用いてゲーム木を表している．

- 行動戦略の標準形ゲームに翻訳したときの均衡

```
?- nash(behavior_strategy(g50(weak)),Players,A,P), Players=[1,2].

A = [a1, [ (a1->a2), (b1->a2)]]
P = [0, 2] ;

A = [b1, [ (a1->a2), (b1->b2)]]
P = [0.5, 0] ;

A = [a1, [ (a1->b2), (b1->a2)]]
P = [0, 2] ;

A = [b1, [ (a1->b2), (b1->b2)]]
P = [0.5, 0] ;

No
```

- 最初の 2 つの均衡の部分ゲーム完全性を検証する

```
?- subgame_perfect(g50(weak), Players,A,P), Players=[1,2].

trial([0, 2], [a1, a2], [a1, [ (a1->a2), (b1->a2)]])
  subgame(player:1/[1, 0], [a1, [ (a1->a2), (b1->a2)]], [0, 2])ne
  subgame(player:2/[0, 2], [a1, [ (a1->a2), (b1->a2)]], [0, 2])ne
  subgame(player:2/[0, 2], [b1, [ (a1->a2), (b1->a2)]], [0.5-1, -1])
  defeated_by([[b1, [ (a1->a2), (b1->b2)]], [0.5, 0]])
trial([0.5, 0], [b1, b2], [b1, [ (a1->a2), (b1->b2)]])
  subgame(player:1/[1, 0], [b1, [ (a1->a2), (b1->b2)]], [0.5, 0])ne
  subgame(player:2/[0, 2], [a1, [ (a1->a2), (b1->b2)]], [0, 2])ne
  subgame(player:2/[0, 2], [b1, [ (a1->a2), (b1->b2)]], [0.5, 0])ne

A = acts([b1, [ (a1->a2), (b1->b2)]])
P = payoffs([0.5, 0])

Yes
```

4. 知識水準でのゲームモデリング

Aumann[Aumann 76]は，共通知識(common knowledge)を初めて定式化し，不一致に合意しえないという直観をきちんと証明した．Milgrom と Stokey は Aumann の結果を一般化し，合理的取引の不可能性の結果を示した．以下では Milgrom と Stokey[Milgrom 82]の例に沿って，3 種類の知識水準での取引者とその推論を Prolog によってモデリングする．詳しくは確率や条件付期待値の扱いを含むソースコード(trade.pl)およびレビュー記事 [Indo 03]を参照されたい．

Players＼state Partition＼	s1	s2	s3	s4	s5
H1	a	b	b	c	c
H2	x	x	y	y	z

図 3 パーティション情報構造の例(Milgrom and Stokey,1982)

4.1 情報構造と共通知識

状態の有限集合Ω，事象の集合 2^{Ω}，取引者の集合をN={1,2}とする．各プレイヤーi の情報構造 H_i は，Ωの非空部分集合すなわち情報集合を値とする状態 s∈Ωの関数 $H_i(s) \subseteq \Omega$，$\phi \neq H_i(s)$として定義される．図 3 は Milgrom と Stokey の例題における取引者の情報構造を表す．この例題のように，各プレイヤーの情報構造がΩのパーティションである場合をパーティション情報構造という．Prolog では例えば次のように書ける．

- 情報構造

```
partition(1,s1,[s1]).
partition(1,S,[s2,s3]):-member(S,[s2,s3]).
partition(1,S,[s4,s5]):-member(S,[s4,s5]).
partition(2,S,[s1,s2]):-member(S,[s1,s2]).
partition(2,S,[s3,s4]):-member(S,[s3,s4]).
partition(2,s5,[s5]).
```

それぞれ状態 s における各取引者の知識は $K(s)E \Leftrightarrow H_i(s) \subseteq E$，また共通知識は $CK(s)E \Leftrightarrow M(s)=\cup_{s \in E} H_i(s)$ $(\forall i \in N)$, $M(s) \subseteq E$ と定義される．つまり共通知識は共通部分を有する情報集合の合併操作の収束先である M(s)の下での知識である．またこれはプレイヤーが互いに可能と考えうる状態を列挙することに等しい．図 3 の情報構造で M(s)はつねにΩに達する．

- 相互推論

```
think(J,S,is_possible(O)):- partition(J,S,H), member(O,H).
think(J,S,K):- K = think(_J1,O,_O1),think(J,S,is_possible(O)),K.
test_think(S,S1,X):- X=[S1,S2,S3,S4,S5,S6],
   think(1, S1,
    think(2, S2,
     think(1, S3,
      think(2, S4,
       think(1, S5,
        think(2, S6,
         is_possible(S)
        )
       )
      )
     )
    )
   ).
```

- 実行例

```
?- test_think(s5,s1,C).
C = [s1, s1, s2, s3, s4, s5]
Yes
```

4.2 合理的な取引者

図 3 の情報構造を仮定し，2 名の取引者が交代手番で取引意向を表明する過程をシミュレーションする．取引が行われた場合の各取引者の勝率などは図 4 のようであるとする．

State	p(w1)	p(w2)	p(w2|s)
s1	.20	.05	1/5
s2	.05	.15	3/4
s3	.05	.05	1/2
s4	.15	.05	1/4
s5	.05	.20	4/5

図 4. 取引ゲームの勝率(Milgrom and Stokey,1982)

取引が成立するためには，任意の時点以降，両者の意向がつねに一致しなければならない．合理的な取引者は自分のパーティションから，相手のパーティションや相手が想像する自分のパーティションについて推理する．ただし各取引者は自分の勝率が 1/2 を越えないと取引しようとしないものとする．

- モデル 1:素朴な取引者

```
trader(naive,J,S,Q,D):-
    partition(J,S,H),
    win_prob_on_event(J,H, Q),
    decision(Q,D).
decision(Q, ok):- Q > 0.5.
decision(Q, reject):- Q < 0.5.
decision(Q, indifferent):- Q = 0.5.
```

- モデル 2:浅読みの取引者

```
trader(sophist,J,S,Q,reject):-
   trader(naive,J,S,Q,reject);
   trader(naive,J,S,Q,indifferent).
trader(sophist,J,S,Q,ok):-
   trader(naive,J,S,Q,ok),
   partition(J,S,H),
   \+ ( member(S1,H), agent(J1), J1 \= J,
        trader(naive,J1,S1,_Q1,reject)).
```

- モデル 3:合理的期待

```
trader(rational,J,(T,S),Q,D):-
   state(S), time(T/N), T > 0,
   agent(J), move(J,T/N,yes),
   delay(T,1,T1),   %T1 is T - 1,
   partition(J,T1/N,S,H),
   win_prob_on_event(J,H, Q),
   decision(Q,D).
```

最初の 2 タイプの取引者は静的パーティションである．合理的な取引者(モデル 3)は，各時点で相手からの取引の申し出の可能性を再推論し，不可能な状態を削除することにより，動的にパーティションを更新する．シミュレーションによって，理論どおり，モデル 3 の取引者はどの状態でも取引に応じないことが確かめられるが，過渡的には合意するケースがあることも分かった．また例題の情報構造を非パーティションに変更することにより，取引発生を確認できる．詳細は[Indo 03]で述べている．

5. おわりに

Prolog は定理自動証明研究から派生したプログラミング言語であり，期待どおり，比較的簡単な各種のゲーム分析とプレイヤーの推論を表せた．ゲーム木の扱いなどになお改良の余地があるが，今後，ゲームデザインとインテリジェンスの関係を理解するために役立てたい．

参考文献

[Aumann 76] Aumann, R. J.: Agreeing to disagree, Annals of Statistics 4: 1236-1239 (1976).

[Imai 01] 今井晴雄・岡田章ゲーム理論の新展開，勁草書房 (2001).

[Indo 02] Indo, K.: Implementing Nash implementation theory with Prolog: A logic programming approach, 第 6 回実験経済学コンファレンス予稿(2002)

[Indo 03] Indo, K.: Common knowledge , mimeo (2003), http://www.us.kanto-gakuen.ac.jp/indo/kw/ck03b.html.

[Milgrom 82] Milgrom, P. and N. Stokey: Information, trade and common knowledge, Journal of Economic Theory 26: 17-27 (1982).

[Muto 01] 武藤滋夫: ゲーム理論入門，日経文庫 (2001).