macc 97

マルチエージェント相互作用としての“じゃんけん”

中島秀之
電子技術総合研究所

概要

Interaction is an important key to cooperative computation. It is known that mutual belief cannot be established by exchanging one way communications. Executing *janken*, a Japanese way of tossing a coin, in a fare manner using only one way communications is also difficult. In this paper, we explore the possibility of using *time* to do janken by exactly two message passing, one for each participants. The purpose is not in janken itself, but in the study of nature of multiagent communication.

1 何がしたいか

協調計算（未定義用語．“三人寄れば文殊の知恵”[中松本 96]）にとっては相互作用が重要である．相互作用の一つにエージェント間の通信がある．

理論的には一方向通信の組合せで共通知識は達成できない [HM90]．

Halpern and Moses は共有知識形成の不可能性を示すための例として一斉攻撃問題を使った．一斉攻撃問題：

> 谷をはさんだ両側に同盟軍が二分割されている（A と B とする）．谷には敵（E）がいる．A と B で同時攻撃を仕掛ければ E を撃退することができるが，A あるいは B 単独では負けてしまう．A と B の間には片方向の通信手段（例えば伝書鳩の往復）しかなく，この通信回路には 100% の信頼性はない．このとき A 軍と B 軍の間に攻撃時刻に関する共有知識は形成できず，攻撃は不可能である．

各軍は，相手が出撃する時刻を知ると同時に，自分がそれを知っているということを相手が知っていることを確信している必要がある．更に自軍にこの確信があることを相手がしらなければ相手は出撃してこないので，この確信を相手が知っていることを確信している必要がある．これは無限に続くが，有限の，届くことが確実ではない片方向通信の連鎖ではこれは達成できない．

しかし人間は共通知識があるがごとく振舞っている．この可能性としては

1. なんらかの両方向通信を使っている．
2. 人間にも共通知識はない．

のどちらかである．

じゃんけんはある種の共通知識が達成されていると考えられる例であると考えられる．（ただしどういう共通知識であるのかは未だに明確ではない．少なくともじゃんけんの手は，相手に伝わった保証が得られず共通知識とはならない．）

ここではプログラムのじゃんけんを考慮することにより上記の問題を堀下げたい．

2　じゃんけん成立の必要条件

後出しがシステムとして不可能か，あるいは後出しが可能であってもそれが検出できること．

3　双方向通信

XP[FF86].　システムがメッセージの同時交換を保証しているので，後出し不能．しかし，XP をネットワーク上に実現することを考えると，これは概念的にも実装的にもサーバ方式と同じである．

4　片方向通信

手を決めたことは確認できるが，中身は読めない状態が必要．この状態を相互知識とすれば良い．

4.1　暗号方式

暗号を用いたじゃんけん [岡 97]：

1. 手を暗号化して相手に送る．
2. 相手からの暗号を受けとる．
3. 鍵を相手に送る．

この方式では 2 回の通信が必要である．また，2 種類あるいは 3 種類の復号が可能な暗号を用意すれば，相手の手を解読してから，それに勝つ / 引き分ける鍵を送信することにより，後出しと同等のことが可能である．

4.2　時間による同期

相手には自分の手を見てから書き直す時間がないようにする．

もし，書き込みに必要な時間が読み出しより有意に長いならば．．．　(ほぼ) 同時に書き始め，直ちに読み込みキューに入る．相手の手を読んだ時刻から相手が書き始めた時刻を計算し，それが自分の書き終えた時刻より先であれば，相手は後出ししていない．

同期が失敗した場合にはやり直す．

もし，読み出しに必要な時間が書き込みより有意に長いならば．．．　書いたらすぐ読み始める．読み出しに必要とされる時間より前に相手の手が届けば，それは後だしではないことが保証される．

実際，人間のじゃんけんはこの原理 (すなわち，人間の反射速度が遅いので相手の手を見てから自分の手を変えられない) に依存しているように思われる．

以下ではこのアイデアを発展させる．

読み出しにかかる最低時間を保証する一つの方法は，ある程度の計算時間をかけないと値を知り得ないような関数を用意することである．例えば，ロジスティック関数

$$x_{n+1} = a x_n (1 - x_n)$$

は $a \geq 3.5699$ の領域でカオスが生じることが知られている．例えば a=4.0 として，x_0 と十分大きな n を与え，$3x_n \mod 3$ を自分の手とすれば良い．あるいは，a もパラメータとしても良い．

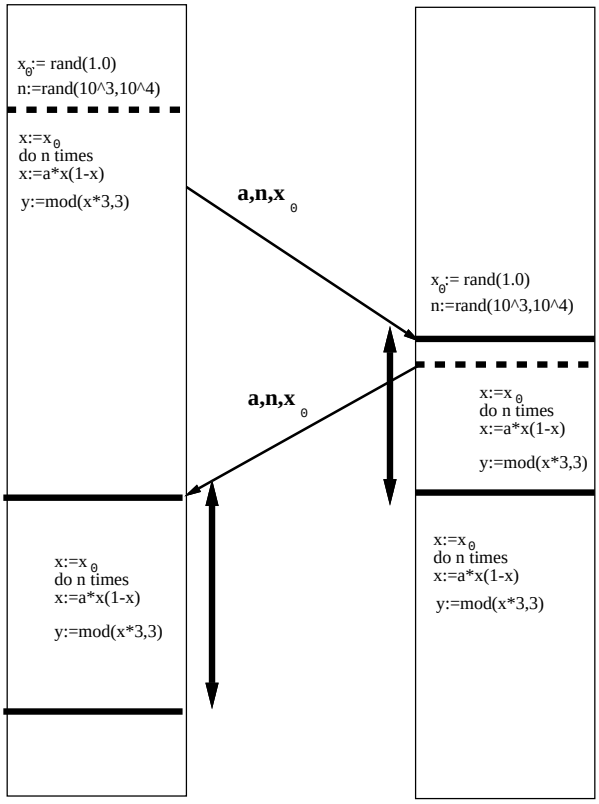

図 1: 時間遅れを利用したじゃんけん

この手法は以下のような特徴を持つ：

1. 逐次計算以外の手段で値を予測できない（カオス）．
2. n はいくらでも大きくできる．
3. 逐次計算しか手段がないので並列マシンでも最低計算時間は有利にならない．
4. 初期値 (必要とあれば a も) はいくらでも変化させられるので，あらかじめ表を作っておくことは不可能．近似解もない．

5 結論

サーバを用いない，1 回通信によるじゃんけんの手法を提案した．この方式は時間遅れを巧妙に用いている．この結果は共有知識の形成に用いることはできず，Halpern and Moses の論文の結論を変えるものではない．

しかしながら“時間”の概念の導入によりマルチエージェント間の相互作用の可能性に幅が持てるのではないかと考えている．

参考文献

[FF86] Robert E. Filman and Daniel P. Friedman. *Coordinated Computing – Tools and techniques for distributed software.* （邦訳：協調型計算システム –分散型ソフトウエアの技法と道具立て–，マグロウヒル）, 1986.

[HM90] Joseph Y. Halpern and Yoram Moses. Knowledge and common knowledge in a distributed environment. *JACM*, 37:549–587, 1990.

[岡 97] 岡本龍明. インターネットでゲームを公平に行なう仕組み. 日経エレクトロニクス, 3.24(685):203–208, 1997.

[中松本 96] 中島秀之, 松原仁, and 本位田真一, editors. 協調プログラミング例題集. 共立出版, 1996. bit 別冊.