Universal Games of Incomplete Information

John H. Reif
Computer Science Department
The University of Rochester
Rochester, New York 14627

ABSTRACT

We consider two-person games of incomplete information in which certain portions of positions are private to each player and cannot be viewed by the opponent. We present various games of incomplete information which are universal for all reasonable games. The problem of determining the outcome of these universal games from a given initial position is shown to be complete in doubly-exponential time. We also define "private alternating Turing machines" which are alternating Turing machines with certain tapes and portions of states private to universal states. The time and space complexity of these machines is characterized in terms of the time complexity of deterministic Turing machines, with single and double exponential jumps.

We also consider blindfold games, which are restricted games in which the second player is not allowed to modify the common position. We show various blindfold games to have exponential space complete outcome problems and to be universal for reasonable blindfold games.

We define "blind alternating Turing machines" which are private alternating Turing machines with the restriction that the universal states cannot modify the public tapes and public portion of states. A single exponential jump characterizes the relation between the space complexity of deterministic Turing machines.

1. Introduction

A (generalized two-person) game G consists essentially of disjoint sets of positions for two players named 1 and 2, plus relations specifying legal next-moves for the players. A position Π contains portions which are private to each player (invisible to their opponent) and the remaining portions of Π are common and may be publicly viewed by both players. The set of legal next-moves for a given player must be independent of the opponent's private portions of positions.

The game G is of perfect information if no position contains a private portion. On the other hand, a game is blindfold if player 2 never modifies the common portion of a position.

This work was supported in part by Alfred P. Sloan Foundation grant # 74-12-5.

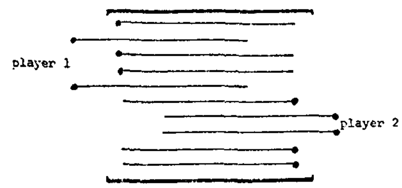


Figure 1a: A position of PEEK

For example, consider the game PEEK of Figure 1a. (Peek was first described in [Chandra and Stockmeyer, 1976].). A position of PEEK consists of a box with two open ends and containing various plates stacked horizontally within. The plates are perforated by holes of uniform size in various places. The top and bottom of the box are also perforated with holes. Each plate contains a knob on either of the open ends of the box, and the plate may slide to either of two locations "in" or "out". Once "out", a plate can only be pushed "in", and vice versa. The players stand at the two open ends of the box. A move by a player $a \in \{1,2\}$ consists of grasping a knob from his side and pushing the corresponding plate either "in" or "out". The player may also pass. If just after the move there is a hole in each plate lined up vertically (so the player can "peek" through from the top), then player a wins.

The game PEEK is of perfect information: each player knows the pattern of holes on the plates and can view the location of all the plates.

To introduce private portions of positions, we place partial barriers on both ends of the box, as in Figure 1b. These barriers hide the location of some, but perhaps not all, of the opponent's plates. Nevertheless, both players are still aware of the pattern of holes in the plates and can attempt to "peek" through the box from the top.

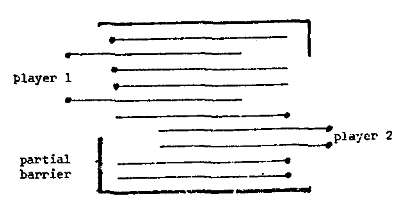


Figure 1b: A position of PRIVATE-PEEK

Let PRIVATE-PEEK be the resulting game of incomplete information. By requiring that the barriers on the side of player 1 obscure the locations of all the opponent's plates, we have the blindfold game BLIND-PEEK.

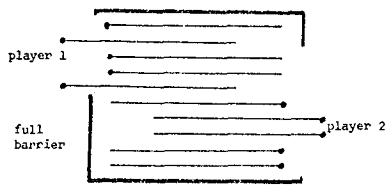


Figure 1c: A position of BLIND-PEEK

The outcome of a game G is the problem of determining the existence of a winning strategy for player 1, given an initial position Π_0. Let the reach of a game G be the function of n giving the maximum number of positions reachable from a given position of length n

If no apriori-bound is placed on the reach of games, the outcome problem is undecidable (see the computation games of section 3).

We define in section 2 a class of games which are reasonable in the sense that (1) the next-move relations are polynomial time computable, and (2) the number of positions reachable from a given initial position Π is bounded by an exponent in the size of Π.

Given a class of games $\mathcal{C}$, a game G^U is universal to $\mathcal{C}$ if (1) $G^U \varepsilon \mathcal{C}$ and (2) the outcome problem for each $G \varepsilon \mathcal{C}$ is log-space reducible (see [Stockmeyer and Meyer, 1973]; a log-space reduction is always polynomial time) to the outcome problem for G^U.

The game PEEK was shown universal to reasonable games of perfect information in [Chandra and Stockmeyer, 1976]. We show BLIND-PEEK is universal for all blindfold reasonable games, and that PRIVATE-PEEK is universal for all reasonable games.

While the outcome problem for PEEK is (log-space) complete in exponential time, the outcome problem for BLIND-PEEK is complete in exponential space, and the outcome problem for PRIVATE-PEEK is complete in double exponential time.

Games (with easy-to-compute next-move relations) can be considered to be computing machines. Game G accepts input ω, considered to be a position of G, depending on the outcome of the game from ω. Games of perfect information are similar to the alternating Turing machines (A-TMs) of [Chandra and Stockmeyer, 1976] in which existential states (identified with player 1) alternate with universal states (player 2) during a computation.

In this paper we introduce the notions of "private alternation" and "blind alternation". In "private alternation" we add to an A-TM certain portions of states and certain work tapes private to universal states (player 2); the machine cannot read the private tapes while in existential states. The result is a PA-TM. In "blind alternation" we restrict a PA-TM so that the universal states can write only on their private tapes, and on no other tapes. The resulting machine is a BA-TM. Acceptance of input strings by these machines is defined by the outcome in corresponding computation games.

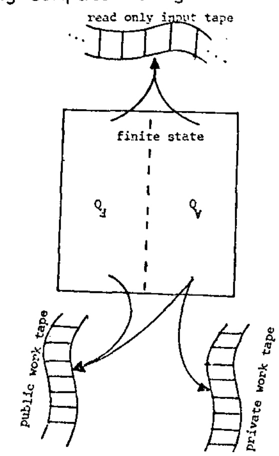


Figure 2: An alternating Turing machine with a tape private to the universal states $Q_\forall$

Let F(n) be a set of functions on variable n. For each $\alpha \varepsilon \{D,A,PA,BA\}$, let

αSPACE[F(n)](αTIME[F(n)])

be the class of languages computable by α-TMs within some space(time) bound in F(n). Also let EXP(F(n)) be the set of functions

$\{c^{f(n)} \mid c > 0 \text{ and } f \varepsilon F(n)\}$

and let EXP(f(n)) denote EXP({f(n)}).

[Chandra and Stockmeyer, 1976] relate the space and time complexity of A-TMs and D-TMs as follows:

For each $S(n) \geq \log n$,

ASPACE[S(n)] = DTIME[EXP(S(n))]

ATIME[EXP(S(n))] = DSPACE[EXP(S(n))]

We characterize the time and space complexity PA-TMs and BA-TMs in terms of the time and space complexity of A-TMs and D-TMs as follows:

For each $S(n) \geq \log n$,

BASPACE[S(n)]= ATIME[EXP(S(n))]
= DSPACE[EXP(S(n))]

PASPACE[S(n)]= ASPACE[EXP(S(n))]
= DTIME[EXP(EXP(S(n)))]
= PATIME[EXP(S(n))]

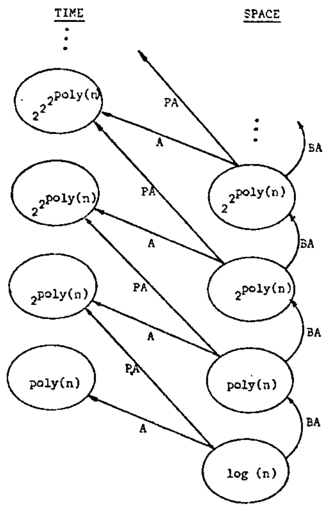


Figure 3: Complexity Jumps for α-TMs from α-SPACE to deterministic time and space.

α = A for "alternation"
= BA for "blind alternation"
= PA for "private alternation"

This paper is organized as follows: the next section defines games of incomplete information, section 3 introduces our "private alternating" and "blind alternating" Turing machines, section 4 presents our complexity results, section 5 described certain propositional formula games which are universal for reasonable games, section 6 concerns universal blindfold games on finite state automata, section 7 concerns pursuit games and in section 8 we conclude this paper with an open problem concerning probabilistic strategies.

2. Combinatorial Games of Incomplete Information

Let a (two-person) game be a tuple

$$G = (P_1, P_2, \underset{1}{\rightarrow}, \underset{2}{\rightarrow}, CP, PP_1, PP_2)$$

where CP, PP_1, PP_2 are sets,

$$P_1, P_2 \subseteq CP \times PP_1 \times PP_2, \ \underset{1}{\rightarrow} \subseteq P_1 \times P_2,$$

and $\underset{2}{\rightarrow} \subseteq P_2 \times P_1$.

The players are named 1, and 2 and are considered to be opponents of each other. Fix a player $a \in \{1,2\}$ and let b be his opponent. A position Π of player a is a tuple $(\pi_c, \pi_1, \pi_2) \in P_a$ where

(1) π_1 is the portion of Π private to player 1 ,

(2) π_2 is the portion of Π private to player 2,

(3) π_c is the portion of Π shared in common by both players.

Let $\text{visible}_a(\Pi) = (\pi_c, \pi_a)$; intuitively, this is the portion of position Π that player a may view. Also, let $\text{private}_a(\Pi) = \pi_a$.

The next-move relation $\underset{a}{\rightarrow}$ contains the set of pairs $(\Pi, \Pi') \in P_a \times P_b$ such that player a has a legal move from position Π to position Π'. We require that $\underset{a}{\rightarrow}$ be independent of that portion of a position private to player b. Formally, if $\Pi \underset{a}{\rightarrow} \Pi'$ then $\text{private}_b(\Pi) = \text{private}_b(\Pi')$. Also, if $\Pi \underset{a}{\rightarrow} \Pi'$ and $\overline{\Pi} \underset{a}{\rightarrow} \overline{\Pi}'$ and $\text{visible}_a(\Pi) = \text{visible}_a(\overline{\Pi})$, then $\Pi \underset{a}{\rightarrow} \overline{\Pi}'$ and $\overline{\Pi} \underset{a}{\rightarrow} \Pi'$. A position $\Pi \in P_a$ with no next move for player a is winning for player b and losing for player a.

Fix an initial position $\Pi_0 \in P_a$.

The pair $G_{\Pi_0} = (G, \Pi_0)$ is a concrete game.

Concrete game G_{Π_0} is

(1) perfect information if $\text{private}_1(\Pi)$, $\text{private}_2(\Pi)$ are empty for each $\Pi \in P_{\Pi_0}$ (i.e. no position $\Pi \in P_{\Pi_0}$ contains a position with a portion private to any player).

(2) blindfold if $\Pi \underset{2}{\rightarrow} \Pi'$ implies $\text{visible}_1(\Pi) = \text{visible}_1(\Pi')$ for all $\Pi \in P_{\Pi_0}$ (i.e. player 2 never modifies the common portion of a position).

(3) solitaire if $\Pi \underset{2}{\rightarrow} \Pi'$ and $\Pi \underset{2}{\rightarrow} \Pi''$ implies $\Pi' = \Pi''$ for each $\Pi, \Pi', \Pi'' \in P_{\Pi_0} - \{\Pi_0\}$ (i.e. the next-move for player 2 is uniquely defined for all positions of player 2 in $P_{\Pi_0} - \{\Pi_0\}$).

A game G is perfect information (blindfold, solitaire) if each of its concrete games are. Chess, Checkers, and Go are all concrete games of perfect information. Some examples of blindfold games are given in Section 6. Also see [Jones, 1978].

Kriegspiel, the German game of "blind chess" is not truly a concrete blindfold game since there is a gradual transfer of positional knowledge as the game progresses. Battleship and Mastermind are (for this same reason) not concrete blindfold games, but are concrete solitaire games.

A play of G_{Π_0} is a (possible infinite) sequence $\Pi_0, \Pi_1, \Pi_2, \Pi_3, \ldots$ of positions of player a alternating with positions of player b such that $\Pi_0 \underset{a}{\rightarrow} \Pi_1$, $\Pi_1 \underset{b}{\rightarrow} \Pi_2$, $\Pi_2 \underset{a}{\rightarrow} \Pi_3$, ...

A play is winning for player $a \in \{1,2\}$ if it is finite and terminates with a win for player a.

Let P_{Π_0} be the set of positions reachable from Π_0, and including Π_0. The game tree of G_{Π_0} is the minimal (but possible infinite) directed tree T_{Π_0} with node set N, root $n_0 \in N$, and node labelling $\Pi : N \rightarrow P_{\Pi_0}$ such that for each play Π_0, Π_1, Π_2, ... there is a path $n_0, n_1, n_2, \ldots$ with $\Pi(n_i) = \Pi_i$ for $i = 0,1,2,\ldots$. The node set N is partitioned into sets N_1, N_2 consisting of those nodes labelled in P_1, P_2 respectively.

For each player $a \in \{1,2\}$, we define an equivalence relation $\underset{a}{\sim}$ over N as follows:

(1) $n_0 \underset{a}{\sim} n_0$,

(2) for all nodes $n, m \in N - \{n_0\}$ with parents n', m' , let $n \underset{a}{\sim} m$ if and only if $n' \underset{a}{\sim} m'$ and $\text{visible}_a(\Pi(n)) = \text{visible}_a(\Pi(m))$,

(3) no other nodes are related by $\underset{a}{\sim}$.

Intuitively, $\underset{a}{\sim}$ relates those nodes of N which player a cannot distinguish in plays of G_{Π_0}.

Let $a, b \in \{1,2\}$ be distinct players.

Let N'_a be the set of nodes of N_a with at least one sibling.

A (deterministic) strategy for player a is a mapping $\sigma : N'_a \rightarrow N_b$ such that for all $n, m \in N'_a$,

(1) $\sigma(n)$ is a sibling of node n,

(2) $n \underset{a}{\sim} m$ implies $\sigma(n) \underset{a}{\sim} \sigma(m)$.

Player a plays by strategy σ if $\Pi(n_0)$, $\Pi(n_1)$, $\Pi(n_2)$, ... is a play and $n_i \in N'_a$ then $n_{i+1} = \sigma(n_i)$. The strategy σ is winning if player a wins on all maximal plays by strategy σ. Note that a strategy for player a essentially defines a mapping from the visible portions of plays (ending with a position of player a) to some legal next-move for player a. Winning strategies for games of perfect information can always be made Markov strategies, i.e. can be made independent of previous play and only dependent of the current position. Winning strategies for games of incomplete information, on the other hand, often depend highly on previous play to determine (at least partly) the private position of the opponent.

Winning strategies are now characterized by finite subtrees of the game tree T_{Π_0}. The subtree $T_{\Pi_0,\sigma}$ induced by strategy σ is derived from the game tree T_{Π_0} by deleting each subtree rooted at m and the edge leading to m, for all nodes $m \in N_b - \{n_0\}$ not in the range of σ.

Proposition Strategy σ is winning for player a if and only if

(1) $T_{\Pi_0,\sigma}$ is finite but not empty,

(2) all leaves of $T_{\Pi_0,\sigma}$ are labelled with winning positions for player a.

(Winning strategies may also be characterized by minimal fixed points of functions on strategies.)

The outcome of concrete game G_{Π_0} is a win for any player $a \in \{1,2\}$ with a winning strategy, else is a draw.

The outcome problem for game G is: given initial position Π_0, determine if player 1 has a winning strategy.

Let game G' be derived from game G by making common to both players that portion of positions originally private to player 1. Note that the outcome problem for game G' is identical to the outcome problem for the original game G. (Nevertheless, the outcome of probabilistic strategies, as defined in the final section, are highly dependent on the existence of private positions of both players.).

The game $G = (P_1, P_2 \underset{1}{\rightarrow}, \underset{2}{\rightarrow}, CP, PP_1, PP_2)$ is reasonable if

[1] each position $\Pi \in P_1 \cup P_2$ is represented as a string in $\{0,1\}^*$,

[2] if position Π' is reachable from position Π by a play of G_Π, then $|\Pi| = |\Pi'|$.

[3] for each player $a \in \{1,2\}$, the pairs of the next-move relation $\underset{a}{\rightarrow}$ are recognizable in polynomial time.

Note that any concrete reasonable game can be realized as a physical object (i.e., with a finite game board and finite sets of tokens for marking positions). A game satisfying only assumptions [1] and [3] is essentially a universal computing machine, as formalized in the next section.

3. Alternating Automata with Private Tapes

The alternating automata proposed by [Chandra & Stockmeyer, 1976] have a natural correspondence to games of perfect information. The states of alternating automata are named either universal or existential. The sequencing between existential and universal states corresponds to the alternation of moves by players in the play of a game. In particular, the outcome problem for reasonable games of perfect information is log-space equivalent to the recognition problem for linear-space bounded alternating Turing machines.

We introduce here the notion of alternating automata with private tapes and private portions of states, which have a natural correspondence to games of incomplete information. In fact, we will define the languages accepted by these machines by the existence of winning strategies for the corresponding computation games. The outcome problem for reasonable games of incomplete information is log-space equivalent to the acceptance problem for linear-space bounded alternating Turing machines with private tapes.

Let an alternating Turing machine with private tapes (PA-TM) be a 12-tuple

$$M = (Q_c, Q_p, Q, q_0, Q_F, Q, \Sigma, \Gamma, \#, b, t_c, t_p, \delta)$$

where

Q_c are the common portions of states

Q_p are the private portions of states

$Q \subseteq Q_c \times Q_p$

Q is the set of states of M

$q_0 \in Q$ is the initial state

$Q_F \subseteq Q$ are the universal states

($Q_\exists \equiv Q - Q_\forall$ are the existential states)

Σ, Γ are the sets of input and tape symbols

$\#, b \in \Gamma - \Sigma$ are end marker and blank symbols

$\delta \subseteq (Q \times \Gamma^t) \times (Q \times \Gamma^t \times \{\text{left}, \text{right}\}^t)$ is the next-move relation, where $t = t_c + t_p + 1$.

There is a single, read-only input tape (named 0) initially containing $\#\omega\#$, where $\omega \in \Sigma^*$ in the input string. There are also $t_c + t_p$ work tapes, initially containing two-way infinite strings of the blank symbol b. The t_c common work tapes $1, \ldots t_c$ might be read or written on from any state of Q, whereas the t_p private work tapes $t_c+1, \ldots, t_c+t_p$ can only be read and written from the universal states of Q. Also, δ must be independent of the private portions of existential states and must not contain state transitions from existential states in which the private portion of the state is modified.

The PA-TM is a natural generalization of machines previously described in the literature. If M has no private tapes, it is an alternating Turing machine (A-TM) as described by [Chandra & Stockmeyer, 1976].

If M is further restricted to only existential states, then it is a <u>nondeterministic Turing machine</u> (<u>N-TM</u>) as is now common in the literature. If M is still further restricted to be deterministic, then we have a <u>deterministic Turing machine</u> (<u>D-TM</u> or just <u>TM</u>), the common, garden-variety universal machine as originally envisioned by Turing.

We now define still another machine (this machine will be relevant to blindfold games.)

Let a <u>BA-TM</u> be a PA-TM restricted so that the universal states never write on the common and input tapes, never move their heads, and never modify the common portion of a state.

Let a <u>configuration</u> be a sequence

$$C = (x_0 q y_0, \ldots, x_{t_c+t_p} q y_{t_c+t_p})$$

such that x_j, y_j are the non-blank prefixes of tape j to the left and right of the scan head, and $q \in Q$ is a state. Configuration C is <u>existential</u> (<u>universal</u>, <u>accepting</u>) if state q is.

Let $NEXT_M(C)$ be the set of configurations which are reached from configuration C by a single step of M, as defined by the relation δ.

We now define the computation game

$$G_M = (P_1, P_2, \underset{1}{\rightarrow}, \underset{2}{\rightarrow}, CP, PP_1, PP_2)$$

where

- P_1 is the set of existential configurations of M,
- P_2 is the set of universal configurations of M,
- CP are the input tape and common work tape portions of configurations of M.
- PP_1 is empty,
- PP_2 are the private work tape portions of configurations of M.

In the computation game G_M, player 1 is identified with the existential states and player 2 is identified with the universal state.

Given configuration C as above with $q = (q_c, q_p)$, let

$$\text{visible}_1(C) = (x_0 q_c y_0, \ldots x_{t_c} q_c y_{t_c}).$$

Thus, $\text{visible}_1(C)$ consists of those portions of configuration C containing the contents of the input tape, the common work tapes, and the common portion of the state.

For each $C_1 \in P_1$ and $C_2 \in P_2$, let

(1) $C_1 \underset{1}{\rightarrow} C_2$ if and only if $C_2 \in NEXT_M(C_1)$,

(2) $C_2 \underset{2}{\rightarrow} C_1$ if and only if $C_1 \in NEXT_M(C_2)$.

Note that if M is a BA-TM with universal configuration C, then

$$\text{visible}_1(C_1) = \text{visible}_1(C_2)$$

for all $C_1 C_2 \in NEXT_M(C)$.

Let $\omega \in \Sigma^*$ be an <u>input string</u>. The initial configuration C_0 is considered an initial position of the concrete computation game $G_{M,C_0} = (G_M, C_0)$. We introduce some terminology to aid the reader's intuition.

Each play of G_{M,C_0} is a <u>computation sequence</u> and the game tree T_{C_0} is a <u>computation tree</u>. The input string ω is <u>accepted by M</u> if there is a winning strategy σ for player 1 in the concrete computation game G_{M,C_0}. The corresponding subtree $T_{C_0,\sigma}$ induced by σ is an <u>accepting subtree</u> of T_{C_0}.

It is easy to verify that PA-TMs accept precisely the recursive enumerable sets. The next section considers the computational complexity of time and space bounded PA-TMs.

4. <u>Complexity of Alternating Turing Machines with Private Tapes</u>

We wish to characterize the time and space complexity of PA-TMs and BA-TMs in terms of the time and space complexity of A-TMs and TMs. As we develop these characterizations, we will describe their applications to reasonable games. Let $\omega \in \Sigma^n$ be a string input to a PA-TM M and let C_0 be the corresponding initial configuration of M.

<u>M accepts ω within time $t \geq 0$</u> if there exists an accepting subtree (of the computation tree) which is of depth $\leq t$.

<u>M accepts ω within space $s \geq 0$</u> if there exists an accepting subtree with no configuration containing a work tape in which more than s cells have been visited.

<u>M accepts language $L \subseteq \Sigma^*$ within time $T(n)$ (space $S(n)$)</u> if M accepts exactly the strings of L within time T(n) (space S(n)).

M has time(space) bound $f(n)$ if M accepts string $\omega \in \Sigma^n$ if and only if M accepts ω within time (space) $f(n)$.

For α = PA, BA, A, N, and D we use the notation αTIME(T(n)) (αSPACE(S(n))) to denote the class of languages accepted by α-TMs within time $T(n)$ (space $S(n)$).

The fundamental results (i.e., tape reduction, constant factor "speed up", complexity hierarchies) for complexity classes of time and space bounded TMs hold also for PA-TMs. (These will be described in detail in a later draft of this paper.)

The following two Theorems, due to [Chandra and Stockmeyer, 1976]

(1) characterize the space complexity of A-TMs in terms of the time complexity of TMs, and

(2) bound the time complexity of A-TMs in terms of the space complexity of N-TMs and TMs.

Theorem 1 For all $S(n) \geq \log n$

$$\mathrm{ASPACE}(S(n)) = \bigcup_{c>0} \mathrm{DTIME}(c^{S(n)})$$

Theorem 2 For all $T(n) \geq n$ and $S(n) \geq n$,

(a) $\mathrm{ATIME}(T(n)) \subseteq \mathrm{DSPACE}(T(n))$

(b) $\mathrm{NSPACE}(S(n)) \subseteq \mathrm{ATIME}(S(n)^2)$.

Let G be a game.

Let $reach_G$ be a function over the natural integers such that for $n \geq 0$,

$reach_G(n)$ is the maximum number of distinct positions reachable from any position Π of G with $n = |\Pi|$.

Observe that for any reasonable game G there exists a constant $c > 0$ such that $reach_G(n) \leq c^n$ for all $n \geq 0$.

Note that for each PA-TM M with space bound $S(n)$, the computation game G_M has $reach_{G_M}(n) = c^{S(n)}$ for some $c > 0$.

In section 3, the outcome of G_M was used to define the language of M. Similarly, for each game G, with $reach_G(n) \geq n$, there is a $\log(n)^k \cdot \log(reach_G(n))$-space bounded PA-TM whose language precisely characterizes the outcome of G, (where k is a constant related to the degree of the polynomial bounding the time complexity of the next-move relations of game G.)

These correspondences also hold

(1) between blindfold games and BA-TMs, and

(2) between games of perfect information and A-TMs.

As a consequence of Theorem 1,

(a) the outcome problem for any reasonable game G of perfect information with $reach_G(n) \geq n$ can be solved within time $reach_G(n) \cdot g(n)$, for some polynomial $g(n)$.

($g(n)$ is related to the time complexity of the next-move relations of G.)

(b) for any $S(n)$, $\log n \leq S(n) \leq n$, there exists a computation game $G_{S(n)}$ which is universal in the class of reasonable games of perfect information with reach $\leq c^{S(n)}$; this game $G_{S(n)}$ of perfect information has outcome problem log-space complete in $\mathrm{DTIME}(c^{S(n)})$. In particular, there exists a computation game G_{PI} (i.e., that associated with linear-space bounded A-TMs) universal for the class of reasonable games of perfect information, and with outcome problem log-space complete in

$$\mathrm{EXPTIME} = \bigcup_{c>0} \mathrm{DTIME}(c^n).$$

We shall derive analogous results for games of imcomplete information and the more restrictive blindfold games.

Let M be a standard PA-TM if

(1) for all configurations C,C' such that $C \in \mathrm{NEXT}_M(C')$ C is universal if and only if C' is existential.

(2) the initial state is existential.

(3) there is a unique accepting state q_A and no next move from q_A.

Lemma 1 For each $S(n) \geq \log n$,

$$\mathrm{PASPACE}(S(n)) \subseteq \bigcup_{c>0} \mathrm{ASPACE}(c^{S(n)})$$

Proof

Let M be a standard PA-TM with space bound $S(n) \geq \log n$. Given input string $\omega \in \Sigma^n$, let C_0 be the initial configuration. Let $d \geq 0$ be the number of distinct symbols occurring in configurations of M. Assume temporarily that $S(n)$ is constructable. The following program runs in space $O(d^{S(n)})$, on an A-TM.

ALGORITHM A

[1] $t \leftarrow 0$; $\mathcal{C}_\exists \leftarrow \{C_0\}$; $C_\exists \leftarrow C_0$

[2] $\mathcal{C}_\forall \leftarrow \{C' \in NEXT_M(C) \mid C \in \mathcal{C}_\exists\}$

[3] if $\mathcal{C}_\forall = \emptyset$ then reject

[4] if all elements of $\mathcal{C}_\forall$ are accepting then accept

[5] existentially choose some $C_\forall \in NEXT_M(C_\exists)$

[6] universally choose some $C_\exists \in NEXT_M(C_\forall)$

[7] $\mathcal{C}_\exists \leftarrow \{C'' \in NEXT_M(C') \mid C' \in \mathcal{C}_\forall$, $visible_1(C') = visible_1(C_\forall)$, and $visible_1(C'') = visible_1(C_\exists)\}$.

[8] if $\mathcal{C}_\exists = \emptyset$ or $t > d^{d^{S(n)}}$ then reject

[9] $t \leftarrow t+2$; go to [2].

It is easy to verify by induction on variable t that the above program accepts if and only if M does. Note that if S(n) is not constructable, then we try $S(n) = \log(|\omega|), \log(|\omega|)+1, \ldots$ until ω is accepted. □

Applying Theorem 1,

$$ASPACE(c^{S(n)}) = \bigcup_{d>0} DTIME(d^{c^{S(n)}})$$

and thus we have:

Theorem 3 For each $S(n) \geq \log n$,

$$PASPACE(S(n)) \subseteq \bigcup_{\substack{c>0\\d>0}} DTIME(d^{c^{S(n)}})$$

As a consequence of Theorem 3, the outcome problem for any reasonable game G with $reach_G(n) \geq n$ can be solved in deterministic time $d^{reach_G(n) \cdot g(n)}$ for some $d > 0$ and polynomial $g(n)$.

We now consider blindfold games.

Lemma 2 For each $S(n) \geq \log n$,

$$BASPACE(S(n)) \subseteq \bigcup_{c>0} NSPACE(c^{S(n)})$$

Proof

Let M be a standard BA-TM. Given input $\omega \in \Sigma^n$ with corresponding initial configuration C_0, we apply only a slight modification of Algorithm A. Recall that in a BA-TM, the universal states can modify neither the common work tapes nor the common portions of configuration. Thus for any universal configuration $C_\forall$, if $C_1, C_2 \in NEXT_M(C_\forall)$, then $visible_1(C_1) = visible_1(C_2)$.

This implies that for our machine M we can optimize Algorithm A by replacing statement [6] with the statement:

[6'] deterministically choose some $C_\exists \in NEXT_M(C_\forall)$.

The result, Algorithm A', runs in nondeterministic space $O(c^{S(n)})$. □

The containment relation

$$NSPACE(c^{S(n)}) \subseteq DSPACE(c^{2S(n)})$$

is implied by Theorem 2 and is due to [Savitch, 1970]. We have established:

Theorem 4 For each $S(n) \geq \log n$,

$$BASPACE(S(n)) \subseteq \bigcup_{d>0} DSPACE(d^{S(n)}).$$

Consequently, for each reasonable blindfold game G of $reach_G(n) \geq n$, the outcome problem for G may be solved in deterministic space $O(reach_G(n)\log(n)^k)$ for some $k > 0$.

In order to lower bound the space complexity of BA-TMs and PA-TMs, it is useful to define an extension of regular expressions with which we can compactly define accepting computations of deterministic TMs and A-TMs. We then show that BA-TMs and PA-TMs can determine the relevant language properties of these expressions in small space.

Let f(n) be a function on the natural integers. Let a power (f(n))-extended regular expression be a regular expression R augmented with an operation for taking powers of the form $(-)^p$ where p is an integer $\leq f(n)$ and n is the size of the expression R. Let R be simple is no power is taken over a subexpression containing a power. (Note that a simple power (2^n) extended regular expression R with $n = |R|$ can be expanded to a (non-simple) power (2)-extended regular expression R' of size $\leq n^2$.)

Lemma 3 For each simple power (f(n))-extended regular expression R with alphabet Σ and length n, there is a BA-TM M with space bound $O(\log n + \log(f(n)))$ which accepts if and only if $L(R) \neq \Sigma_1^*$.

Proof

Consider first the case where R contains no powers. There is an obvious BA-TM M in which a string $x_1, x_2, \ldots x_k \in \Sigma_1^*$ is existentially constructed, symbol by symbol, and the universal states attempt to show $x_1, x_2, \ldots x_k \in L(R)$.

The universal states keep a private pointer to the currently considered subexpression of R.

This pointer is stored in log n cells of a work tape private to the universal states.

In the case R contains subexpressions of the form $(-)^p$, and R is simple, the universal states of M must also store a counter of size $\leq f(n)$ on $\log(f(n))$ cells of their private work tape. Thus M accepts if and only if $L(R) \neq \Sigma^*$. □

Next we give an obvious extension of results of [Mayer and Stockmeyer, 1973] for power(2)-extended regular expressions. (The entire proof is given here only because some of the details will be crucial to results later in the section.)

Lemma 4 Let M be a nondeterministic TM with space bound f(n) with c>0 and $f(n) \geq n$. Given any input string $\omega \in \Sigma^*$, there is a simple power (f(n))-extended regular expression R over alphabet Σ_1 such that M accepts ω if and only if $L(R) \neq \Sigma_1^*$.

Furthermore, R is of size O(n).

Proof Let Q be the set of states of M and let $q_0, q_A \in Q$ be the initial and final states.

Let Γ be the tape symbol alphabet with blank sumbol $b \in \Gamma$ and $\# \notin \Gamma \cup (Q\times\Gamma)$. We consider pairs $[q,a] \in (Q\times\Gamma)$ as distinct symbols. Using the usual tape reduction techniques we can assume M has but one tape.

Let a configuration C be represented as a string of length f(n) of the form $\alpha[q,a]\beta$ where $\alpha,\beta \in \Gamma^*$, $q \in Q$ is the current state and $a \in \Gamma$ is the currently scanned tape symbol. Given an input string $\omega = \omega_1 \omega_2 \ldots \omega_n \in \Sigma^*$ let the initial configuration be $C_0 = [q_0,\omega_1]\omega_2\ldots\omega_n b^{f(n)-n}$.

Let an accepting computation be a sequence $\#C_0\#C_1\#C_2\ldots\#C_k$ where

[P1] C_0 is the initial configuration.

[P2] $C_i \in NEXT_M(C_{i-1})$ for $i=1,2,\ldots,k$.

[P3] C_k contains the accepting state q_A.

Let $\Sigma_1 = \Gamma \cup (Q\times\Gamma) \cup \{\#\}$

Let $R_1 = ((\Sigma_1-\#) +\#\cdot((\Sigma_1-[q_0,\omega_1])+\omega_1\cdot((\Sigma_1-\omega_2)$

$+ \omega_2\cdot((\Sigma_1-\omega_3)+ \ldots \cdot(\Sigma_1-\omega_n)))\ldots)\cdot\Sigma^*$

$+ \Sigma_1^{n+1}\cdot b^*\cdot(\Sigma_1-b-\#)\cdot\Sigma_1^*$

$+ \#\cdot(\Sigma_1+\lambda)^{f(n)-1}\cdot\#\cdot\Sigma_1^*$

$+ \#\cdot\Sigma_1^{f(n)}-(\Sigma_1-\#)\cdot\Sigma_1^*$

Note that $\Sigma^*-L(R_1)$ is the set of strings with prefix of the form $\#C_0\#$ where C_0 is the initial configuration.

For each $a_{-1}, a_0, a_1 \in \Sigma_1$, if $a_0=\#$ then let $F(a_{-1},\#,a_1)=\#$ and otherwise let $F(a_{-1},a_0,a_1) = \{a' \in \Sigma_1 |$ if a_{-1},a_0,a_1 are the i-1,i,i+1 symbols of string #C#, then a' is the ith symbol of the string #C'#, where $C' \in NEXT_M(C)\}$.

Let $R_2 = \bigcup_{a_{-1}a_0a_1\in\Sigma_1} (\Sigma_1^*\cdot a_{-1}\cdot a_0\cdot a_1\cdot\Sigma_1^{f(n)-1}\cdot(\Sigma_1-F(a_{-1},a_0,a_1)).$

and note that $\Sigma_1^* -L(R_3)$ is a set of strings that satisfy property P2.

Finally, let $R_3 = (\Sigma_1-(\bigcup_{a\in\Gamma}[q_A,a]))^*$

and note that $\Sigma_1^* -L(R_2)$ is a set of strings containing the accepting state.

Thus we have a simple power (f(n))-extended regular expression $R = R_1+R_2+R_3$ such that $x \in L(R)$ if and only if x is not an accepting computation. □

As a consequence of Theorem 4 and Lemmas 3 and 4, we have:

Theorem 5 For each $S(n) \geq \log n$,

$$\text{BASPACE}(S(n)) = \bigcup_{d>0} \text{DSPACE}(d^{S(n)}).$$

(Note in our simulation of a $f(n) = d^{S(n)}$ space bounded deterministic TM by a S(n)-space bounded BA-TM, in the case $S(n) < n$ we do not actually construct the power(f(n))-extended regular expression R of Lemma 4, but instead keep enough information on private work space (O(log n) space is sufficient for this) to "virtually construct" R from the input string ω i.e. construct that part of R which is needed at a given time in the simulation.)

We now extend the above proof technique to PA-TMs.

Lemma 5 For each $S(n) \geq \log n$, $\bigcup_{c>0}\text{ASPACE}(c^{S(n)}) \subseteq \text{PASPACE}(S(n))$

Proof Let M be an A-TM with space bound $f(n) = c^{S(n)}$, for some c>0. We can assume M satisfies the various restrictions required of the TM in Lemma 4 (except of course M is an A-TM). Fix an input string $\omega \in \Sigma^n$ and let C_0 be the initial configuration.

There is a constant $d_0 \geq 0$ dependent only on M, such that $d_0 \geq |NEXT_M(C)|$ for any configuration C.

For each configuration C with

$NEXT_M(C) = \{C_1, C_2, \ldots, C_k\}$

let $NEXT_M^j(C) = \{C_j\}$ for $j=1,\ldots,k$.

$= \{C_k\}$ for $j=k+1,\ldots,d$.

We now define functions similar to F of Lemma 4.

For each $j=1,\ldots,d$ and $a_{-1}, a_0, a_1 \in \Sigma_1$, if $a_0 = \#$ then let $F^j(a_{-1},\#,a_1) = \#$ and otherwise let $F^j(a_{-1},a_0,a_1) = \{a' \in \Sigma_1 |$ if a_{-1},a_0,a_1 are the $i-1,i,i+1$ symbols of the string #C#, where C is a universal configuration, then a' is the ith symbol of the string #C'#, where $C' \in NEXT_M^j(C)\}$.

Let $R_2^{(j)}$ be the power (S(n))-extended regular expression identical to R except that the function F^j is used in place of F. (Note: Let M^j be the deterministic TM derived from M by requiring that M^j take only the jth branch from a universal state and halt in an existential state. The language of

$\Sigma_1^*-L(R_1+R_2^{(j)}+R_3)$

contains exactly the accepting computations of M^j.) Also, let $R^{(0)} = R$.

We sketch the construction of a PA-TM M_1 which essentially the existential states will construct subtrees of T_{C_0} and the universal states will attempt to show that these subtrees are not accepting subtrees for M.

It will be useful to consider T_{C_0} to be represented as a "branching string" in which each path down T_{C_0} corresponds to a linear string $\#C_0\#C_1\#\ldots$ which is a computation sequence of M. The branches occur just after the symbol #.

An integer J and boolean FLAG are stored in a work tape common to both universal and existential states. Initially J and FLAG are 0. On the private tape we store enough information to easily "virtually construct" $R_2^{(J)}$ from the input string ω, as in the above note. Also on the tape private to the universal states we store a counter for powers in R, as in the proof of Lemma 3.

The machine M_1 is programmed with a minor iteration loop within a major iteration loop. In the minor loop the existential states generate a sequence $x_1,x_2,\ldots,x_k \in \Sigma_1$, symbol by symbol, with the universal states alternatively attempting to discover that $y_1y_2\ldots y_k \; x_1x_2\ldots x_k$ is contained in $L(R_2^{(J)})$, where $y_1y_2\ldots y_k$ are symbols generated by the existential states in the previous major loop. (The operation of M_1 within the minor loop is similar to the BA-TM described in the proof of Lemma 3.)

If any x_j is a state q of M, then FLAG is set to either 0 or 1 depending on whether q is existential or universal. When (if ever) an $x_i = \#$ is chosen, then the machine M_1 leaves the minor loop and sets J to 0 if FLAG = 0 and otherwise universally sets J to some element of $\{1,2,\ldots,d\}$. The machine M_1 next scans over the cells containing J while in existential states. The minor loop is then entered again.

If FLAG = 1, the above steps have the effect of creating a branch in the subtree of T_{C_0} generated by the existential states.

Each branch must succeed, i.e. must be an accepting subtree. Also within the minor loop the universal states attempt to verify that if linear string $z_1,z_2\ldots z_k$ is a path down the "branching string" of symbols generated by the existential states, then $z_1,z_2\ldots z_k \in L(R_1+R_3)$.

Operating in this manner, M_1 accepts if and only if M accepts the input string ω. Note that since the integer J is upper bounded by the fixed constant d, the PA-TM M_1 has within a constant the same space bound as the BA-TM described in the proof of Lemma 3. Since $f(n) = c^{S(n)}$, M_1 has space bound

$O(\log(f(n))) = O(S(n))$. □

As a consequence of Theorems 1, 4, and Lemma 5, we have the primary result of this paper:

Theorem 6 For each $S(n) \geq \log n$,

$$\text{PASPACE}(S(n)) = \bigcup_{\substack{c>0\\d>0}} \text{DTIME}(d^{c^{S(n)}}).$$

Theorems 5 and 6 have important applications to reasonable games.

For each $S(n)$, $\log n \le S(n) \le n$, and $d>0$, (1) there is a computation game $G^{S(n)}$ universal for the class of reasonable games with reach $\le d^{S(n)}$ and the outcome problem of $G^{S(n)}$ is log-space complete in

$$\mathrm{ASPACE}(d^{S(n)}) = \bigcup_{c>0} \mathrm{DTIME}(c^{d^{S(n)}})$$

(2) there is a blindfold computation game $\overline{G}^{S(n)}$ universal for the class of reasonable blindfold games with reach $\le d^{S(n)}$, and $\overline{G}^{S(n)}$ has outcome problem log-space complete in $\mathrm{DSPACE}(d^{S(n)})$.

In particular, there is a game G^U, universal for all reasonable games, and a game G^{UB}, universal for all reasonable blindfold games. G^{UB} has outcome problem log-space complete in

$$\text{EXP-SPACE} = \bigcup_{d>0} \mathrm{DSPACE}(d^n)$$

and G^U has outcome problem log-space complete in

$$\text{EXP-EXP-TIME} = \bigcup_{\substack{c>0\\ d>0}} \mathrm{DTIME}(d^{c^n}).$$

We now consider the time complexity of PA-TMs and BA-TMs.

The following Theorem provides an upper bound to the computational power of time bounded PA-TMs.

tight.

Theorem 7 For each $T(n) \ge n$

$$\mathrm{PATIME}(T(n)) \subseteq \bigcup_{c>0} \mathrm{DTIME}(c^{T(n)})$$

Proof Let M be a standard PA-TM with constructable time bound $T(n)$. Given an input string $\omega\varepsilon\Sigma^n$, let C_0 be the initial configuration. Let T_{C_0} be the computation tree.

Step 1 Let D_{C_0} be the acyclic digraph constructed from T_{C_0} by

(1a) deleting all nodes of T_{C_0} of depth $>T(n)$

(1b) collapsing all remaining nodes n,m such that $n \sim_1 m$ (the equivalence relation $\sim_1$ is defined in Section 2).

Note that D_{C_0} contains no more than $d^{T(n)}$ nodes, for some constant $d \ge 0$ dependent only on M. The step (1b) may be accomplished by a breadth-first search of T_{C_0}, from the root to the nodes of level $T(n)$.

For $t = 1,2,\ldots,T(n)$ collapse together all nodes n,m of level t such that

(a) the parents of n,m have been collapsed together at level $t-1$.

(b) if $C(n),C(m)$ are the configurations associated with nodes n,m then $\mathrm{visible}_1(C(n)) = \mathrm{visible}_1(C(m))$, i.e. the public work tape portions of $C(n)$, $C(m)$ are equal.

Step 2 Next, we prune various nodes and edges from D_{C_0} to form a digraph D'_{C_0}. Repeatedly pass through D, considering each node n which was not derived entirely from accepting configurations.

Delete node n and all entering and departing edges if (1) n was derived from existential configurations and has no departing edges.

(2) n was derived from universal configurations and at least one edge, originally departing from n in D_{C_0} has been deleted.

The resulting digraph D'_{C_0} can be constructed in deterministic time $O(c^{T(n)})$ for some $c>0$. We claim D'_{C_0} is nonempty if and only if M accepts the input string ω within the $T(n)$.

Suppose M accepts ω. Then there is an accepting subtree T'_{C_0} (of the game tree T_{C_0}) with depth $\le T(n)$. If we apply steps (1) and (2) to T'_{C_0}, the result is a non-empty subgraph of D'_{C_0}. Hence D_{C_0} is non-empty.

On the other hand, if D'_{C_0} is nonempty, then the tree derived from D'_C (by separating common descendants of the root) contains an accepting subtree of depth $\le T(n)$ as a subgraph.

If $T(n)$ is not constructable, then we try $T(n) = n,n+1,\ldots$ until acceptance. □

The following Lemma generalizes a divide and conquer technique used by [Savitch, 1970] to show

$$\mathrm{NSPACE}(S(n)) \subseteq \mathrm{DSPACE}(S(n)^2)$$

and used by [Chandra & Stockmeyer, 1976] to show

$$\mathrm{NSPACE}(S(n)) \subseteq \mathrm{ATIME}(S(n)^2).$$

Lemma 6 for each $S(n) \geq n$,

$$\mathrm{ASPACE}(S(n)) \subseteq \mathrm{PATIME}(S(n)^2)$$

Proof Let M be a standard A-TM with space bound $S(n)$ and let $\omega \in \Sigma^n$ be in input string. The algorithm below runs in time $O(S(n)^2)$ on a PA-TM. The variables C_1, C_2, and C_3 are stored on private tapes.

Algorithm B

[1] $C_1 \leftarrow$ the initial configuration C_0.
[2] universally choose to goto [3] or [10].
[3] universally choose some configuration C_2 size $\leq T(n)$.
[4] existentially choose to goto [5] or [7].
[5] exchange the contents of C_2 and C_3
[6] goto [2]
[7] if C_2 is a universal configuration then write C_2 on a common tape
[8] exchange the contents of C_2 and C_1
[9] goto [2]
[10] if $C_3 \in \mathrm{NEXT}_M(C_1)$ then accept else reject

Note that statement [7] forces branching on universal configurations. We can show M rejects ω if and only if Algorithm B accepts. Since $\mathrm{ASPACE}(S(n))$ is closed over complementation, the result follows. □

Theorem 8 For each $T(n) \geq n$,

$$\mathrm{PATIME}(T(n)) = \bigcup_{c>0} \mathrm{DTIME}(c^{T(n)}).$$

Let a game G have time limit $T(n)$ if for each initial position Π winning for player 1, there is an induced subtree of depth $\leq T(n)$, when $n = |\Pi|$. The above Theorem implies that there is a reasonable computation game G^{PTL} with polynomial time limit which has outcome problem log-space complete in $\text{EXP-TIME} = \bigcup_{c>0} \mathrm{DTIME}(c^n)$.

We did not succeed in precisely characterizing the time complexity of BA-TMs in terms of the time complexity of deterministic TMs. However, we do have a characterization in terms of a generalization of the complexity class Σ_2^P of the polynomial time hierarchy of [Stockmeyer, 1973]

Let M be a Σ_k-bounded A-TM if (1) each initial configuration is existential (2) on any path down any accepting subtree, configurations switch from existential to universal or vice versa at most k-times. Let $\Sigma_k^{T(n)}$ be the class of languages accepted by A-TMs which are both $T(n)$-time bounded and Σ_k-bounded.

We can show:

Theorem 8 for each $S(n) > n$,

$$\mathrm{BATIME}(T(n)) \subseteq \Sigma_2^{T(n)} \subseteq \mathrm{BATIME}(cT(n))$$

for some $c>0$.

As a consequence of this Theorem, there is a computation game G^{BPTL} which is universal for reasonable blindfold games with polynomial time limits, and G^{BPTL} has outcome problem log-space complete in

$$\Sigma_2^P = \bigcup_{k>0} \Sigma_2^{n^k}.$$

5. Universal Games on Propositional Formulas

As consequences of Theorem 5 and 6 of the previous section, we have two computation games G^U and G^{BU} such that

(1) G^U is universal for all reasonable games.

(2) G^{BU} is universal for all reasonable blindfold games.

In this section we construct various propositional formula games which are universal for reasonable games. These games and the reductions between them are generalizations of work on games of perfect information in [Chandra & Stockmeyer, 1976].

Boolean variables take on values 1(true) and 0(false). Let a literal be a boolean variable or its negation. Let a propositional formula F be in k-CNF (k-DNF) for if F consists of a conjunction (disjunction) of formulas $F_1, F_2, \ldots F_j$ with each F_i a disjunction (conjunction) of at most k literals.

We now list 3 games on propositional formulas which are universal for all reasonable games. The games G^3 and G^{3B} are essentially the games PRIVATE-PEEK and BLIND-PEEK described in the introduction. Throughout this section we equate player 2 with 0.

(1) Let G^2 be the game in which a position contains a propositional formula $F(X, Y^C, Y^{P0}, Y^{P1}, a, s)$ in 5 CNF form, with X^C, Y^{P0}, Y^{P1} each sequences of variables and a ,s individual variables, plus a truth assignment to its variables. The formula F and the truth assignment to the variables of X, Y^C, a.s are common to both players 1 and 2, but the truth assignment to the variables of Y^{P0}, Y^{P1} are private to player 0.

Player 1 moves by setting a to <u>true</u> and choosing a new truth assignment for the variables of X. Player 0 moves by (a) setting a to false, (b) setting s to the complement of its previous truth assignment, (c) choosing a new truth assignment for the variables of Y^C, Y^{PS}. The formula F is not modified by these moves, except for the changes in the truth assignment to its variables. The loser is the first player whose move yields a truth assignment for which the formula F is <u>false</u>.

(2) Let $\underline{G}^2$ be the game in which each position contains formulas $WIN_1(U,V^C,V^P)$ and $WIN_0(U,V^C,V^P)$ and truth assignments to the sequences of variables of U,V^C,V^P.

The formulas WIN_1 and WIN_0 and truth assignments to variables $U \cup V^C$ are viewed commonly by both players, but the truth assignment to the variables of V^P are private to player 0. Player 1 moves by changing the truth assignment to at most one variable of U, while player 0 moves by changing at most one variable of U^C,U^P. Player $a \in \{0,1\}$ <u>wins</u> if formula WIN_a is <u>true</u> after a move by player a

(3) Let $\underline{G}^3$ be the game in which a position consists of a propositional formula $F'(U,V^C,V^P)$ in DNF form and a truth assignment to the variables of the sequences U,V^C and V^P. The formula F' and truth assignments to the variables of U, V^C are viewed commonly by both players, but the truth assignment to the variables V^P is private to player 0.

Players move as in game G^2. A player wins if after his move the formula is <u>true</u>. To show G' is universal for reasonable games, we consider a linear-space bounded standard PA-TM M with input $\omega \in \Sigma^n$. We encode each configuration C as a bit vector of length $n' = k_1 \cdot n$ (where k_1 depends only on the size of the tape alphabet of M), so that bits $1,2,\ldots,n_c$ are those of $visible_1(C)$ (the portions of C public to both the existential and universal states), and the bits $n_c+1,\ldots,n'$ contain those portions of C private to the universal states.

Using the techniques of [Stockmeyer, 1975], we can construct a propositional formula

$NEXT(Z_1,Z_2,T)$

where Z_1,Z_2,T are sequences of variables of length n',n',k_2 (where k_2 is a fixed constant) and such that:

if Z_1 encodes a configuration C_1 then there exists an assignment to the variables of T such that $NEXT(Z_1,Z_2,T)$ is <u>true</u> if and only if Z_2 encodes some configuration $C_2 \in NEXT_M(C_1)$. The size of NEXT is linear in the input length n.

We introduce new sequences of variables X,Y^C,Y^{P0},Y^{P1} of length m,m,p,p where $m = n_c + k_2$ and $p = n'-n_c$. Let $Y = Y^C.Y^{P0}.Y^{P1}$.

For distinct $s,\bar{s} \in \{0,1\}$, let $NEXT_{1,2}(X,Y)$ be the formula derived from $NEXT(Z_1,Z_2,T)$ by substituting $X(1),\ldots, X(n_c),Y^{Ps}(1),\ldots,Y^{P\bar{s}}(p)$ for Z_1, substituting $Y^C(1),\ldots,Y^C(n_c),Y^{Ps}(1),\ldots, Y^{Ps}(p)$ for Z_2, and substituting $Y^C(n_c+1),\ldots, Y^C(n_c+k_2)$ for T.

Also, let $NEXT_{0,s}(X,Y)$ be derived from $NEXT(Z_1,Z_2,T)$ by substituting $Y^C(1),\ldots,Y^C(n_c), Y^{P\bar{s}}(1),\ldots,Y^{P\bar{s}}(p)$ for Z_1, substituting $X(1),\ldots, X(n_c), Y^{Ps}(1),\ldots,Y^{Ps}(p)$ for Z_2, and substituting $X(n_c+1),\ldots,X(n_c+k_2)$ for T. If we consider player 1 to be identified with the existential states of M and player 0 to be identified with the universal states of M, then for each $a \in \{0,1\}$, $NEXT_{a,s}$ defines legal moves by player a on switch variable $s \in \{0,1\}$.

Now we consider the formula $F(X,Y^C,Y^{P0},Y^{P1},a,s)$

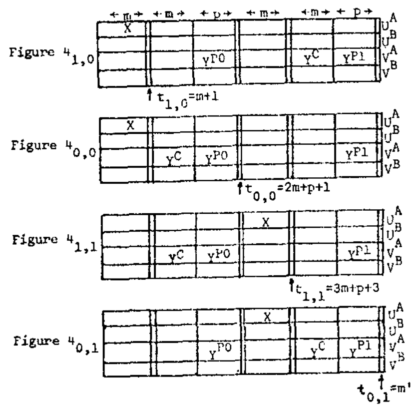
$$= (a \wedge s \rightarrow NEXT_{1,1}(X,Y)) \wedge (a \wedge \neg s \rightarrow NEXT_{1,0}(X,Y)) \wedge (\neg a \wedge s \rightarrow NEXT_{0,1}(X,Y)) \wedge (\neg a \wedge \neg s \rightarrow NEXT_{0,0}(X,Y))$$

F can easily be put in 5 CNF form.

Given the inital configuration C_0 of M on input ω, let the variables $Y^C(1),\ldots,Y^C(n')$ be assigned to encode C_0 and let all other variables be assigned arbitrarily. Let formula F and this truth assignment be an initial position Π_1 of game G^1. Then player 1 wins concrete game $G^1_{\Pi_1}$ if and only if player 1 (the existential states) wins the concrete computation game G_{M,C_0} if and only if M accepts input ω. Thus we have a log-space reduction from the acceptance problem for linear-space bounded M to the outcome problem for G^1, and we conclude that G^1 is universal for reasonable games.

Next, we show the formula game G^2 if also universal for reasonable games.

We now introduce sequences of variables U^A,U^B,V^A,V^B of length $m' = 4m+2p+4$. Let $U=U^A.U^B$ and let $V=V^A.V^B$.



The sequences of variables X,Y defined in the previous construction will, in legal plays of our Game G^2, be contained in U,V as in Figures $4_{1,0}$ $4_{0,0}$, $4_{1,1}$, $4_{0,1}$. The private portion V^P of V is where Y^{P0}, Y^{P1} are located, and V^C contains other elements of V.

For each $s \in \{0,1\}$ and player $a \in \{0,1\}$, let $NEXT'_{a,s}(U,V)$ be the formula derived from formula $NEXT_{a,s}(X,Y)$ by substituting formulas as in Figure $4_{a,s}$.

Again, player 1 (player 0) is identified with the existential (universal) states of M, and $NEXT'_{a,s}$ is used to define next-moves by machine M:

As in a similar construction of [Chandra and Stockmeyer, 1976] we describe a legal play such that if players 1 and 0 play legally, then player 1 wins if and only if M accepts input ω. Let a legal cycle be the following play for $i=1,2,\ldots m'$:

[1] player 1 changes the truth assignment of either $U^A(i)$ or $U^B(i)$

[2] player 0 changes the truth assignment of either $V^A(i)$ or $V^B(i)$.

Consider some $s \in \{0,1\}$ and distinct players $a,b \in \{0,1\}$. Within the legal cycle, for each i, $(t_{a,s} \bmod m')<i \leq t_{b,s}$ player a assigns variables so that $NEXT_{a,s}$ is true when $i = t_{a,s}$. Thus M accepts input ω if and only if player 1 has a winning strategy within legal plays. The following, somewhat tedious construction, forces legal play by both players.

We now introduce some notation for operations on sequences of Z,Z' of boolean variables of length m'. Let $\oplus$ be exclusive-or and let

$$Z \oplus' Z' = (Z(1)\oplus Z'(1),\ldots,Z(n)\oplus Z'(n)),$$

and let

$$\Delta Z=(\neg(Z(n)\oplus Z(1)),Z(1)\oplus Z(2),\ldots,Z(n-1)\oplus Z(n)).$$

Also let $\text{TH-TWO}(Z) = \bigvee_{1\leq i<j\leq m'} (Z(i)\wedge Z(j))$ be the threshold-two function.

To simplify our formulas we define $U'=\Delta(U^A\oplus' U^B)$, and $V'=\Delta(V^A\oplus' V^B)$. which are sequences which locate boundaries between contiguous 0s or 1s.

To detect illegal play we define:

$$ILL_1=\text{TH-TWO}(U')\vee\bigvee_{1\leq i\leq m'} U'(i)\wedge V'(i)\wedge\neg V'(i-1)$$

$$ILL_0=\text{TH-TWO}(V')\vee\bigvee_{1\leq i\leq m'} V'(i)\wedge U'(i+2)\wedge\neg V'(i)$$

as in [Chandra and Stockmeyer, 1976].

Finally, for each distinct players $a,b \in \{0,1\}$, let

$$WIN_a=ILL_b\vee\bigvee_{s\in\{0,1\}} U'(t_{a,s})\wedge V'(t_{b,s})\wedge\neg NEXT'_{a,s}(U,V).$$

Note that formula WIN_0 and WIN_1 can be put in DNF form.

Given input $\omega \in \Sigma^n$, Let π_1 be the initial position of formula game G^1 defined previously.

Let the initial position π_2 of formula game G^2 contain formulas WIN_0, WIN_1 as defined above plus the initial truth assignment of π_1 as in Figure $1_{1,0}$. We consider player 1 wins concrete game $G^2_{\pi_2}$ if and only if M accepts ω. Thus G^2 is also a formula game universal for all reasonable games.

Next, we give a log-space reduction from the outcome problem for formula game G^2 to the outcome problem for formula game G^3.

Let u_0,u_1,u_2,v_0,v_1 be variables not in U or V and set

$$\hat{U} = U\cdot(u_0,u_1,u_2) \text{ and}$$

$$\hat{V}^P = V^P\cdot(v_0,v_1).$$

$$\text{Let } F'(U,V^C,\hat{V}^P)=(WIN_0\wedge(u_0\ v_1))\vee(WIN_1\wedge v_0)\vee(u_1\wedge u_2\wedge\neg v_1)$$

as in [Chandra and Stockmeyer, 1976]. This formula, with initial variable assignments of π_2, form the initial position π_2, of formula game G^3. The concrete game $G^3_{\pi_3}$ is winning for player 1 if and only if $G^2_{\pi_2}$ is winning for player 2. Thus, G^3 is another formula game universal for all reasonable games.

Let G^{2B}, G^{3B} be blindfold games derived from formula games G^2, G^3 by requiring that the common variable sequence V^C of player 0 be empty. We claim that G^{2B}, G^{3B} are universal for all reasonable blindfold games. To show this, we need only note that if M is restricted to a BA-TM, then the universal states never modify the common tape. Hence the common variables V^C in our previous construction contain no information relevant to a configuration of M (though they are useful to insure legal play) and hence the variables V^C may be added to the variables V^P private to player 0.

Note that formula games G^3 and G^{3B} are essentially the games PRIVATE-PEEK and BLIND-PEEK described in the introductory section. Thus we have:

(1) PRIVATE-PEEK is a universal reasonable game.

(2) BLIND-PEEK is a universal reasonable blindfold game.

6. Blindfold Games on Finite State Machines

This section describes two blindfold games G_{FSA} and G_{SUBFSA} which are universal for all reasonable blindfold games of their respective reaches (polynomial and exponential).

Because of simplicity of these games, we feel that they may be useful for establishing the complexity of other blindfold games by log-space reductions. In fact, we apply this technique to pursuit games in the next section.

We consider (nondeterministic) finite state automata (FSA)

$M = (Q, I, F, \Sigma, \delta)$ where

Q is a finite set of states

$I \subseteq Q$ are the initial states

$F \subseteq Q$ are the accepting states

$\delta \subseteq (Q \times \Sigma) \times Q$ is the transition relation.

Let $L(M) \subseteq \Sigma^*$ be the language accepted by M.

Let G_{FSA} be the game in which a position consists of a triple (M,d,q) where $M = (Q, I, F, \Sigma, \delta)$ (the portion of Π common to both players) is a nondeterministic FSA, d (the portion of Π private to player 1) is either 0 or 1, and q (the portion of Π private to player 2) is a state of Q.

Player 1 moves by setting of either to 0 or 1. Player 2 has no next move (player 1 wins) if $q \notin F$ or $\delta(q,d) = \emptyset$ else player 2 moves by replacing q with some state in $\delta(q,d)$. Neither player modifies M, and hence G_{FSA} is a blindfold game.

Note that player 1 has a winning strategy from initial position Π if and only if

$L(M) \neq \{0,1\}^*$.

The problem $L(M) \neq \{0,1\}^*$ was shown by [Stockmeyer and Mayer, 1973] to be log-space complete in P-SPACE = $\bigcup_{k\geq 0} \mathrm{DSPACE}(n^k)$.

Since $\mathrm{reach}_G(n) = n$, we can determine the outcome of G_Π in P-SPACE by applying Algorithm A' of Theorem 4.

Thus, the outcome problem for G is log-space complete in P-SPACE. This result is due to [Jones, 1978], who showed the outcome of a graph reachability game G of $\mathrm{reach}_G(n) = O(n^2)$ was P-SPACE complete.

We now extend this result to a game of exponential reach with outcome problem log-space complete in EXP-SPACE = $\bigcup_{c>0} \mathrm{DSPACE}(c^n)$.

Let a subroutining finite state automata (SFA) $M = (M_0, M_1, \ldots, M_k)$ where for $i = 0,1,\ldots,k$ M_k is a nondeterministic FSA over alphabet $\{0,1\} \cup \{a_j | j>i\}$ where $0,1,a_1,a_2,\ldots,a_k$ are distinct symbols. We assume the empty string is not in the language of $M_1,\ldots,M_k$.

Let the language accepted by M by

$L(M) = \{\omega' \in \{0,1\}^* | \omega'$ be derived from $\omega \in L(M_0)$ by repeatedly substituting a string of $L(M_i)$ for each symbol a_i appearing in $\omega\}$. Note that SUBFSAs accept exactly the languages over $\{0,1\}$ generated by context-free grammars in which no nonterminals appear recursively.

Let G_{SUBFSA} be the blindfold game in which a position Π consists of a triple $(M,d,(q_0,q_1,\ldots,q_k))$ where $M = (M_0,M_1,\ldots,M_k)$ is a SUBFSA common to both players with $M_i = (Q_i, I_i, F_i, \Sigma_i, \delta_i)$, $d \in \{0,1\}$ is private to player 1, and the states $(q_0,q_1,\ldots,q_k) \in Q_0 \times Q_1 \times \ldots Q_k$ are private to player 2.

Player 1 moves by setting d to either 0 or 1. Player 2 has no next move (player 1 wins) if $q_0 \notin F$.

The legal moves of player 2 are defined by the following nondeterministic program

[1] assign d to a temporary d'

[2] nondeterministically choose to go to either [3] or [4]

[3] replace q_0 with some element of $\delta_0(q_0,d')$ and exit

[4] nondeterministically choose some $i,j,q'_j \in Q_j$ such that $1 \le i < j \le k$, $\delta(q_i,a_j) \neq \emptyset$, and $q'_j \in \delta(q_j,d')$

[5] if $q'_j \notin F_j$ then go to [7]

[6] nondeterministically choose to go to [7] or [8]

[7] replace q_j with q'_j and exit

[8] replace q_j with some element of I_j

[9] $d' \leftarrow a_j$

[10] go to [2].

Note that we can test in polynomial time whether a pair of positions of G_{SUBFSA} are a legal move of player 2.

Again, player 1 has a winning strategy from initial position Π if and only if $L(M) \neq \{0,1\}^*$. Also, the outcome of G_{SUBFSA} may be determined by Algorithm A of Theorem 3 in

$$\text{EXP-SPACE} = \bigcup_{c>0} \text{DSPACE}(c^n).$$

Let R be a power(2)-extended regular expression over {0,1}, with a squaring operator $(-)^2$. This squaring operator allows R to concisely define long strings; [Stockmeyer and Mayer, 1973] showed that the problem $L(R) \neq \{0,1\}^*$ is log-space complete in EXP-SPACE. In log-space we can construct a SUBFSA M with the same language as R. Thus, the outcome problem for G_{SUBFSA} is log-space complete in EXP-SPACE. (Note that this result can be viewed as an alternative proof of Theorem 5, restricted to linear-space bounded BA-TMs.)

7. Games of Pursuit

The classical literature of game theory [Blaquiere, 1973; Isaacs, 1965; von Neumann, 1953] contains numerous examples of discrete games of pursuit and evasion; perhaps the popularity of such games is derived from their obvious military applications.

We define in this section a general class of such pursuit games. Our results on the complexity of the outcome of our pursuit games seem quite discouraging to war gamers (but perhaps encouraging to pacifists). Let f_1, f_2 be linear or sublinear functions over the natural integers. Let BLIND-PURSUIT $(f_1(n),f_2(n)) = (P_1,P_2,\vdash_1,\vdash_2,CP_1,PP_1,PP_2)$ where for each position $(\pi_c,\pi_1,\pi_2) \in P_1 \cup P_2$,

(i) a triple (C_s,LOCAL) containing
  (a) a finite degraph $D = (N,E)$
  (b) a node $s \in N$
  (c) a set $\text{LOCAL} \subseteq N$

(ii) π_1 is a multiset of $f_1(|N| + |E|)$ nodes of N

(iii) π_2 is a multiset of $f_2(|N| + |E|)$ nodes of N.

For each $a \in \{1,2\}$, let

$$\text{JUMP}_a = \{ v \in N \mid (u,v) \in E,\ u \in \pi_a \}.$$

We require that $\text{LOCAL} = \text{JUMP}_1 \wedge \text{JUMP}_2$.

Intuitively, player 1 has various "mice" placed on nodes of V, player 2 has various "cats" placed on nodes of V, both players have common knowledge of the digraph D, a node $s \in V$ known as the "mouse hole", and the nodes LOCAL (which may be detected, say, by the animal's sense of smell.) The goal of player 1 is to place some mouse on the "mouse hole" $s \in N$, and the goal of player 2 is to place a cat on a mouse.

Players $a \in \{1,2\}$ move by "jumping" each of their pieces across an edge of E, (by replacing each $n \in \pi_a$ with a node m such that $(n,m) \in E$.)

Player 1 is not allowed to place a mouse on a cat. Player 1 has no next move (player 2 wins) if

$\pi_1 \wedge \pi_2 \neq \emptyset$ (a cat is placed on the same node as a mouse) or

JUMP_1-LOCAL is empty (for some mouse, each possible jump lands on a cat). Player 2 has no next move (player 1 wins) if

$s \in \pi_1$ (some mouse lands on the mouse hole s).

Let PURSUIT($f_1(n)$,$f_2(n)$) be the game of <u>perfect information</u> which is identical to BLIND-PURSUIT($f_1(n)$,$f_2(n)$), except each player is allowed to view his opponent's position. The outcome problem for a game similar to PURSUIT(1,1) (1 mouse vs. 1 cat) was shown log-space complete in

$$\text{P-TIME} = \bigcup_{k\geq 0} \text{DTIME}(n^k)$$

by [Chandra and Stockmeyer, 1976]. We can show the outcome problem for PURSUIT(1,n) is (1 mouse vs. n cats) log-space complete in

$$\text{EXP-TIME} = \bigcup_{c>0} \text{DTIME}(c^n)$$

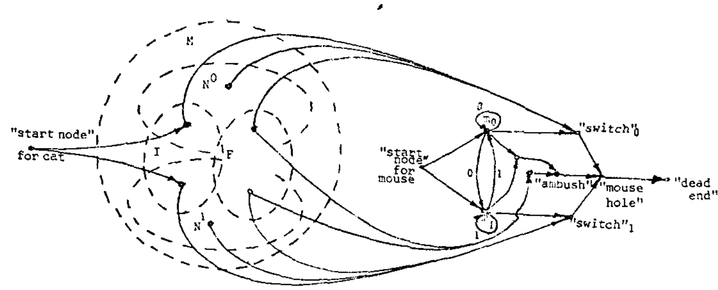
using a log-space induction for the game of PEEK of [Chandra and Stockmeyer, 1976].

We wish now to establish lower bounds on the complexity of BLIND-PURSUIT(1,n) (1 blind mouse vs. n cats). There <u>may</u> exist a log-space reduction for the game BLIND-PEEK (shown in Section 5 to be log-space complete in EXP-TIME) to BLIND-PURSUIT(1,n) but we have not discovered such a reduction.

Instead, we apply the techniques developed in the previous section for blindfold games. Note that BLIND-PURSUIT(1,n) is <u>not</u> a blindfold game since we have allowed players to partially view their opponent's position through the common set LOCAL. Dropping the set LOCAL from the common portions of positions results in a blindfold pursuit game with an easy-to-compute outcome.

We first show the outcome problem for BLIND-PURSUIT(1,1) is P-SPACE hard, and then generalize this reduction to show BLIND-PURSUIT(1,n) is EXP-SPACE hard. Let M be a nondeterministic FSA. We shall represent M as a digraph (N,E) with edge labelling $EL: E \to \{0,1\}$, initial states $I \subseteq N$, and final states $F \subseteq V$, so that $w \in L(M)$ if and only if there exists a (possibly empty) path $(u_0,u_1,\ldots u_\ell)$ of (N_R,E_R) such that $u_0 \in I$, $u_\ell \in F$, and $w = EL(u_0,u_1)\ EL(u_1,u_2)\ \ldots\ EL(u_{\ell-1},u_\ell)$.

It will be useful to assume N_R may be partitioned into two disjoint sets N_0,N_1 such that for each $d \in \{0,1\}$ and $v \in N_R^d$, all edges of E entering v are labelled with d. (N,E) forms a subgraph of our constructed digraph D_M of Figure 5, on which we play our pursuit game.



<u>Figure 5</u>: A 1 blind mouse vs. 1 cat pursuit game in graph D_M derived from FSA M. There is an edge from the "start node" for the cat to each node of I. For simplicity, the edges of M are not illustrated. There is an edge from each node of N_0 to "switch"$_0$. Similarly, there is an edge for each node of N_1 to "switch"$_1$. Finally, there is an edge from each node of F to "ambush".

Each play can be divided into 3 sequences: the <u>initial play</u>, the <u>middle play</u>, and the <u>final play</u>.

(1) In the initial play, the mouse moves from its start node to either of nodes m_0 or m_1. The cat responds with a move from its start node to a node of I.

(2) The middle play lasts until either the cat leaves the nodes of V or the mouse departs from nodes $\{m_0,m_1\}$.

Note that during the middle play, for each $d \in \{0,1\}$, if the mouse moves to m_d, then the cat must respond with a move to a node in V^d, else on the next move the mouse detects that the cat is not in V^d and can dart to the node "switch"$_d$ (which can be immediately reached by the cat only if it is in a node of V^d) and hence the mouse may safely arrive at the "mouse hole" s.

(3) If the mouse is to always reach the "mouse hole" s in the end play with no such blunders by player 2, then it must enter the "ambush" node with the cat not in F. This is so just in the case $\lambda \notin L(M)$ and the middle play is empty, or the moves of the mouse in the middle game are $m_{d_1}, m_{d_2}, \ldots m_{d_\ell}$ and $d_1,d_2 \ldots d_\ell \notin L(M)$.

Thus we have shown that the outcome problem for game BLIND-PURSUIT(1,1) is log-space hard in P-SPACE.

Now we generalize the above construction to the game BLIND-PURSUIT(1,n). Let $M = (M_0, M_1, \ldots M_r)$ be a SUBFSA as defined in section 6. Let a_k be the "subroutine call" symbol associated with M_k, for $k = 1, \ldots r$. For each $k = 0,1,\ldots r$, M_k is represented (as in the previous construction) by a digraph (N_k, E_k) with edge labelling $EL_k : E_k \to \{0,1\} \cup \{a_j \mid j > k\}$, initial states $I_k \subseteq V_k$ and final states $F_k \subseteq V_k$. We further assume

(1) no edge of E_k enters a node of I_k or departs from a node of F_k.

(2) for each node $v \in V_k$, all edges of E_k entering v have the same label, and all edges of E_k departing from v have the same label.

(3) each node $v \in N_k$ whose entering edges are labelled with some $a_j \notin \{0,1\}$ has a "twin" $\hat{v}$ with entering edges from the same destination, the same label, and with departing edges also identical to those of v. These assumptions increase the size of M_k by at most a constant factor.

The digraph H_M on which we will play a game of BLIND-PURSUIT(1,n) will contain as subgraphs only slightly modified versions of $M_0, M_1, \ldots M_r$. Let H_M initially consist of the digraph D_M as defined previously (See Figure 5), plus the disjoint digraphs $M_0, \ldots, M_r$. For each $k = 0,1,\ldots r$ we associate a cat k with M_k. In the case $k > 0$, the cat k will be used to implement "subroutining" on symbol a_k. To this end, we add as in Figure 6a a "start node" for cat k and appropriate edges for reinitializing the cat k after "subroutine" calls. In addition, certain edges and two "switch"$_k$ nodes are added to M_k as in Figure 6b to force the cat k to make the appropriate state transitions in response to moves by the mouse on nodes $\{m_0, m_1\}$.

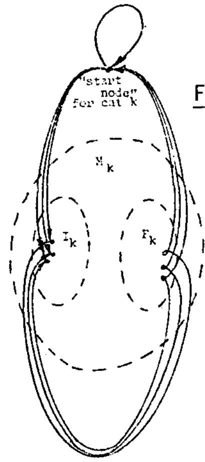


Figure 6a: For each $k > 0$, we add a "start node" for cat k, and an edge from this start node to each node of I_k. To allow reinitialization of cat k after a "subroutine call" on a_k, we add an edge from each node of F_k to the "start node" for cat k and also to each node of I_k. We also add a loop edge at the "start node" for cat k to allow the cat k to wait there until the next subroutine call.

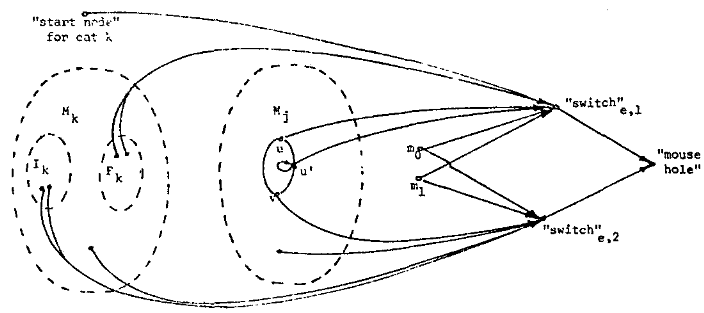


Figure 6b: The following additions to M_k force cat k to take state transitions in M_k over those edges labelled with {0,1} corresponding to moves of the mouse on the nodes $\{m_0, m_1\}$, respectively. For each $d \in \{0,1\}$, let N_k^d be the nodes of M_k which are targets of edges labelled with $d \in \{0,1\}$. (We have assumed all edges entering a node of N_k^d are labelled with d). Add an edge from each node of N_k^d to "switch"$_{k,d}$ for all $d \in \{0,1\}$. These last edges allow the mouse at node m_d to detect whether the cat k is in N_k^d; if not then the mouse can safely jump to the "switch"$_{k,d}$ node and then on to his "mouse hole". Each node v of N_k in neither N_k^0 nor N_k^1 has by assumption a distinct "twin" $\hat{v}$. Add an edge from v to "switch"$_{k,0}$ and an edge from $\hat{v}$ to "switch"$_{k,1}$. These edges insure that the mouse is "blind" to the cases when cat k calls a subroutine and is in neither N_k^0 nor N_k^1.

Now we modify the digraphs $M_0,\ldots,M_r$ to properly implement subroutining:

For each edge $e = (u,v)$ labelled by letter a_k, and contained in M_j, we substitute a small subgraph, and add certain new edges and two "switch"$_e$ nodes as illustrated in Figure 7. These modifications have the effect of stalling cat j at the immediate vicinity of node u while cat k takes a path in M_k from an initial node of I_k. Cat j is only allowed to land on node v when cat k has reached a final node of F_k (note that in this interval, cat k may "call a subroutine" in a similar manner).

During the call, cat k traverses a path in M_k, beginning at a node of I_k. During this time, the cat j must wait at nodes u,u'. When cat k has reached a node of F_k, the cat j moves finally to node v, completing the call to a_k.

We can show, as in the previous construction, that

(1) $\lambda \notin L(M)$ if and only if there is a winning strategy for the mouse in which it reaches its "mouse hole" with an empty middle play, and

(2) $d_1 \ldots d_\ell \notin L(M)$ if and only if there is a winning strategy for the mouse with every middle play a prefix of $m_{d_1},\ldots m_{d_\ell}$.

Hence, the mouse has a winning strategy just in the case $L(M) \neq \{0,1\}^*$. Thus we have that the outcome problem for BLIND-PURSUIT(1,n) is EXP-SPACE hard.

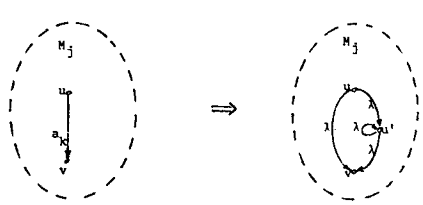


Figure 7a: Modifications of M_j to implement a subroutine call on edge $e = (u,v)$ labelled with a_k. The edge e is replaced with a small subgraph with new node u'.

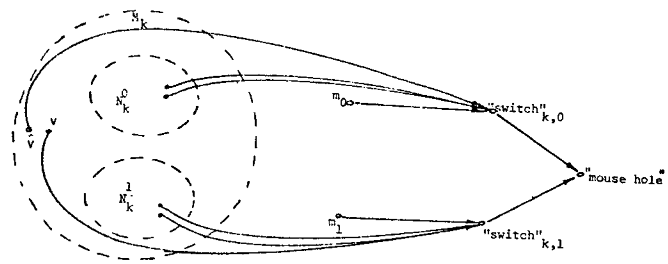


Figure 7b: Additional edges and new "switch" nodes are used to implement the subroutine call of a_k associated with edge $e = (u,v)$ of M_j. We add an edge to "switch"$_{e,1}$ from the "start node" of cat k, the nodes of F_k, and the nodes u,u', m_0,m_1. There is an edge to "switch"$_{e,2}$ from all nodes of M_k but those of F_k, from all nodes of M_j but u and u', and from nodes m_0,m_1.

Just before the call of a_k, we assume the cat j is at node u, the cat k is at its "start node" or in F_k, and the mouse is at m_0 or m_1.

8. Solitaire Games

In section 2 we defined a game

$G = (P_1, P_2, \vec{1}, \vec{2}, CP, PP_1, PP_2)$

to be a solitaire game if given any $\Pi_C \in P_1 \cup P_2$ the next move relation $\vec{2}$ of player 2 in uniquely defined for all positions $\Pi \in P_2 - \{\Pi_0\}$ of player 2 reachable from Π_0. We can show that the outcome problem for reasonable solitaire games is contained within Π_2^P (excuse the conflicting notation) of the polynomial hierarchy [Stockmeyer, 1973], since the outcome problem of any concrete reasonable solitaire game G_{Π_0} can be log-space reduced to the satisfiability problem for a formula of the form

$$\forall x \exists y f_{\Pi_0}(x,y)$$

where f_{Π_0} is a propositional formula in CNF of size polynomial in $|\Pi_0|$.

Furthermore, the satisfiability problem for formulas of the above form can be considered a solitaire game which is universal for all reasonable solitaire games, and is log-space complete in Π_2^P.

9. Conclusion

In [Reif, 1979], we consider the computational complexity of finite state and push-down automata with "private" and "blind" alternation.

We conclude with an open question concerning the computational complexity of the payoff of optimal probabilistic strategies for reasonable games.

Let $G = (P_1, P_2 \vec{1}, \vec{2}, CP, PP_1, PP_2)$ be a game and fix an initial position $\Pi_0 \in P_1 \cup P_2$. Let the game tree be T_{Π_0}, with node set N, edge set E, and root $n_0 \in N$. As usual, with each $n \in N$, there is a corresponding position $\Pi(n)$, with $\Pi(n_0) = \Pi_0$. Let $\Gamma(n)$ be the set of edges of E departing from node $n \in N$. For each player $a \in \{1,2\}$, let $N_a = \{n \in N \mid \Pi(n) \in P_a\}$ be the set of nodes corresponding to positions of player 1. Also, let $E_a = \bigcup_{n \in N_a} \Gamma(n)$ and let

$N'_a = \{n \in N_a \mid \Gamma(n) \neq \emptyset\}$.

A probabilistic strategy for player a is a mapping ρ_a from E_a to the real interval [0,1], such that for all $(n,m),(n',m') \in E_a$, if $n \underset{a}{\sim} m$ and $n' \underset{a}{\sim} m'$ then $\rho_a(n,m) = \rho_a(n',m')$, (i.e. the strategy ρ_a must not vary over edges that player a cannot distinguish) and

$\sum_{m \in \Gamma(n)} \rho_a(n,m) = 1$ for all $n \in N'_a$.

Let ρ_a be deterministic if for each $n \in N'_a$, there exists some $(n,m) \in E_a$ such that $\rho_a(n,m) = 1$. Thus deterministic strategies, viewed as mappings from nodes $n \in N'_a$ to some $m \in \Gamma(n)$, are precisely the sort of strategy defined in section 2.

Let $\underline{S}_1(\underline{S}_2)$ be the class of probabilistic strategies for player 1(2).

Given a pair of probabilistic strategies $\rho_1 \in S_1$ and $\rho_2 \in S_2$, we define for each $n \in N$,

$$\mathrm{PAYOFF}_{\rho_1,\rho_2}(n) = 1 \text{ if } \Pi(n) \text{ is a winning position of player 1;}$$

$$= 0 \text{ if } \Pi(n) \text{ is a winning position of player 2;}$$

and otherwise, if n is not a winning position for either player and $n \in N_a$ then let

$$\mathrm{PAYOFF}_{\rho_1,\rho_2}(n) = \sum_{(n,m) \in \Gamma(n)} \rho_a(n,m) \cdot \mathrm{PAYOFF}_{\rho_1,\rho_2}(m)$$

The probability of player 1 winning, from position Π_0, under an optimal min-max strategy [von Neumann and Morgenstern, 1953] is thus

$$\mathrm{PWIN}(\Pi_0) = \max_{\rho_1 \in S_1} \left(\min_{\rho_1 \in S_2} \left(\mathrm{PAYOFF}_{\rho_1,\rho_2}(n_0)\right)\right)$$

Finally, let

$\mathrm{PWIN}_G = \{(\Pi, \mathrm{PWIN}_1(\Pi)) \mid \Pi$ is a position of game G$\}$.

Note that player 1 has a winning deterministic strategy from initial position Π_0 if and only if $(\Pi_0, 1) \in \mathrm{PWIN}_G$.

Thus the outcome problem (for deterministic strategies) for any game G is trivially reducable of PWIN_G. Let M be a universal TM. Since the outcome problem for computation game G_M is undecidable, PWIN_{G_M} is also undecidable.

What is the complexity of the set PWIN_G for reasonable game G? Theorem 6 of this paper implies that PWIN_G is EXP-EXP-TIME hard for reasonable games G.

Acknowledgements

I would like to thank Paul Spirakis for pointing out the importance of non-Markov strategies in games of incomplete information, and Penelope for interesting me in mouse vs. feline pursuit games.

Bibliography

[1] Blaquiere, A., Ed. Topics in Differential Games, North-Holland Publishing Company, The Netherlands, 1973.

[2] Chandra, A.K. and Stockmeyer, L.J., "Alternation," Proc. 17th IEEE Symp. on Foundations of Computer Science, 1976, pp. 98-108.

[3] Even, S. and Tarjan, R.E., "A combinatorial problem which is complete in polynomial space", 7th Annual ACM Symp. on Theory of Computing, May, 1976, Hershey, pp. 41-49.

[4] Fraenkel, A.S., Garey, M.R., Johnson, D.S., Schaefer, T. and Yesha, Y., "The complexity of checkers on an N×N board-preliminary report," Proc. 19th IEEE Symp. on Foundations of Computer Science, Oct. 1978, pp. 55-64.

[5] Isaacs, R., Differential Games, John Wiley and Sons, Inc., New York, 1965.

[6] Jones, N.D., "Blindfold games are harder than games with perfect information," Bulletin of the European Assoc. for Theoretical Comp. Sci., No. 6, Oct., 1978, pp. 4-7.

[7] Jones, N.D., "Space-bounded reducibility among combinatorial problems," J. Comput. System Sci., 11 (1975), pp. 68-85.

[8] Lichtenstein, D. and Sipser, M., "GO is Pspace Hard," Proc. 19th IEEE Symp. on Foundations of Computer Science, Oct. 1978, pp. 48-54.

[9] von Neumann, J. and O. Morgenstern, Theory of Games and Economic Behavior, 3rd ed., Princeton University Press, 1953.

[10] Reif, J.H., "Private and blind alternating automata," to appear.

[11] Savitch, W.J., "Relationships between nondeterministic and deterministic tape complexities, J. Comput. Sci., 4 (1970), pp. 177-192.

[12] Schaefer, T.J., "Complexity of decision problems based on finite two-person perfect-information games, Proc. 8th ACM Symp. on Theory of Computing, 1976, pp. 41-49.

[13] Stockmeyer, L.J., "The polynomial-time hierarchy, IBM Research Report RC5379, 1975; Theoretical Computer Science, to appear.

[14] Stockmeyer, L.J. and Meyer, A.R., "Word problems requiring exponential time: preliminary report," Proc. 5th ACM Symp. on Theory of Computing, 1973, pp. 1-9.

[15] Stockmeyer, L.J. and Chandra, A.K., "Provably difficult combinatorial games," IBM T.J. Watson Res. Ctr. Research Report, Yorktown Heights, Jan. 1978.