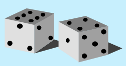
THE CONNECTION BETWEEN GAMES AND COMPUTER SCIENCE



Peter van Emde Boas

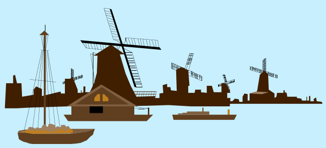
ILLC-FNWI-Univ. of Amsterdam

April 2000

Bejing

&

Xian



References and slides available at: http://turing.wins.uva.nl/~peter/teaching/thmod99.html

Topics

- Computation, Games and Computer Science
- Recognizing Languages by Games; Games as Acceptors
- Understanding the connection with PSPACE (The Holy Quadrinity)
- Interactive Protocols and Games
- Loose Ends in the Model ?

Games in Computer Science

- Evasive Graph properties (1972-74)
- Information & Uncertainty (Traub ea. - 1980+)
- Pebble Game (Register Allocation, Theory 1970+)
- Tiling Game (Reduction Theory - 1973+)
- Alternating Computation Model (1977-81)
- Interactive Proofs /Arthur Merlin Games (1983+)
- Zero Knowledge Protocols (1984+)
- Creating Cooperation on the Internet (1999+)
- E-commerce (1999+)
- Logic and Games (1950+)
- Language Games, Argumentation (500 BC)

COMPUTATION

- Notion of Configurations: Nodes
- Notion of Transitions: Edges
- Non-uniqueness of transition: Out-degree > 1 - Nondeterminism
- Initial Configuration : Root
- Terminal Configuration : Leaf
- Computation : ~~Branch~~ Tree
- Acceptance Condition: Property of trees

Introducing the Opponents

© Games Workshop

© Games Workshop

URGAT
Orc Big Boss

THORGRIM
Dwarf High King

Games involve strategic interaction ......

Bi-Matrix Games

© Games Workshop

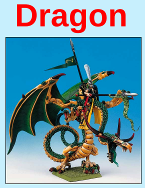
	O	S
R	-1/1	1/-1
D	1/-1	-1/1

© Games Workshop

Runesmith

© Games Workshop

Dragon



© Games Workshop

Ogre

© Games Workshop

Squigg

© Games Workshop

A Game specified by describing the Pay-off Matrix ....

Von Neumann's Theorem

© Games Workshop

	O	S
R	-1/1	1/-1
D	1/-1	-1/1

© Games Workshop

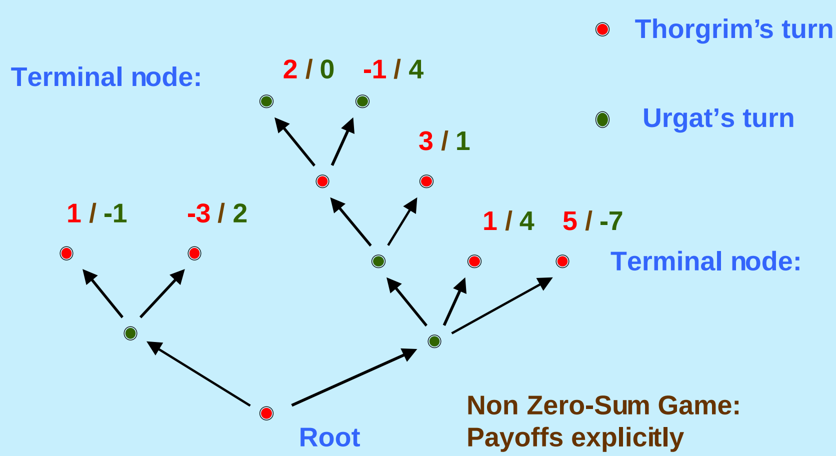
R ( + D )/2 : (O  + S )/2

© Games Workshop © Games Workshop © Games Workshop © Games Workshop

Mixed Strategy Nash Equilibrium;
no player can improve his pay-off by deviation.

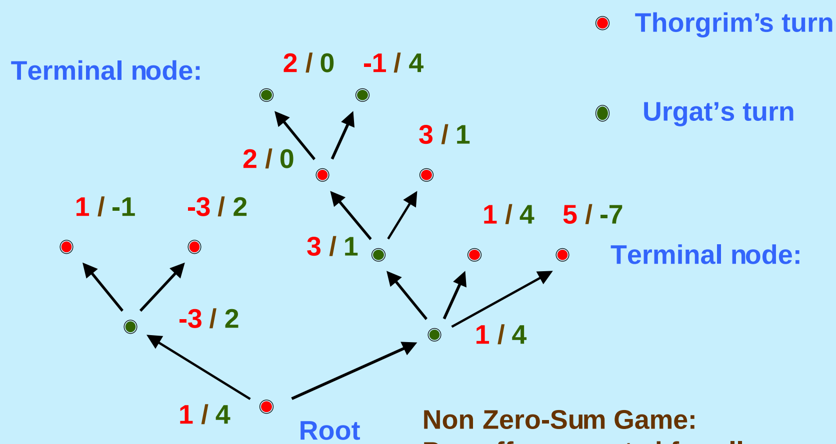
Game Trees

(Extensive Form - close to Computation)



Non Zero-Sum Game:
Payoffs explicitly
designated at terminal node

Backward Induction



Non Zero-Sum Game:
Pay-offs computed for all
game nodes including the Root.

A Game

Starting with 15 matches players alternatively take 1, 2 or 3 matches away until none remain. The player ending up with an odd number of matches wins the game

A Game specified by describing the rules of the game ....

Analysis of the DD game

Extension used:
Thorgrim wins if he has an odd number when the game terminates.
This allows for even n .

Relevant feature: parity of number of matches collected so far (not the number itself!)

Four types of configurations remain:
T/E : Thorgrim has to play and has an even number
T/O : Thorgrim has to play and has an odd number
U/E : Urgat plays, while Thorgrim has an even number
U/O : Urgat plays, while Thorgrim has an odd number

Backward Induction Table

n	U / E	U / O	T / E	T / O
18	U	U	T / 1	T / 2
17	U	T	T / 1	U
16	U	T	U	T / 3
15	U	U	T / 2	T / 3
14	U	U	T / 2	T / 1
13	T	U	U	T / 1
12	T	U	T / 3	U
11	U	U	T / 3	T / 2
10	U	U	T / 1	T / 2
9	U	T	T / 1	U
8	U	T	U	T / 3
7	U	U	T / 2	T / 3
6	U	U	T / 2	T / 1
5	T	U	U	T / 1
4	T	U	T / 3	U
3	U	U	T / 3	T / 2
2	U	U	T / 1	T / 2
1	U	T	T / 1	U
0	U	T	U	T

The Mechanism

Several of the results you hear in the Computation Theory and the Logic and Games Community are of the form:

Formula Φ is OK (true, provable, valid) iff the game G(Φ) has a winning strategy for the first player, where G(Φ) is obtained by some explicit construction.

Topic in these talks: This Mechanism
Which properties can be characterized this way ??

An Unfair Reduction

©Games Workshop

©Games Workshop

If Hoatlacotlincotitli faces an Opponent which is Worthy he will challenge her to a game of HEX where she moves first (and consequently she can win). Otherwise she is the First Player in a game of NIM with piles of sizes 5,6,9 and 10 (which she will loose if Hoatl plays well).

Hence: Only Worthy Opponents have a winning stategy ....

Restrictions are Needed

In the Hoatl scenario the transformation from input to the game is Arbitrary.

We should rather use Polynomial Time Reductions.

Resulting Games must have Polynomial Size Descriptions.

The latter doesn't entail that the resulting Games can be played in Polynomial Time ! (repeating moves can't be excluded....)

The Model: Games as Acceptors

Input X is mapped to some game G(X)

The mapping X à G(X) is easy to compute (computable in Polynomial Time or Logarithmic Space)

Consequence: G(X) has a Polynomial Size Description. (Leaving open what the Proper Descriptions are.)

L_G := { X | G(X) has a winning strategy for the first player }

Which Languages L can be characterized in this way ?

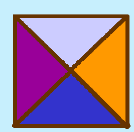
Peter van Emde Boas: The Games of Computer Science, April 2000

Types of Games (and Computations)

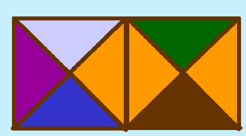
- Single player - no choices : Routine : Determinism
- Single player - choices : Solitaire : Nondeterminism
- Two players – choices : Finite Combinatorial Games : Alternating Computation
- Single player - random moves : Gambling : Probabilistic Algorithms
- Two players - choices and random moves : Interactive Proof Systems
- Several players - concurrent moves : Multi Prover Systems

Tiling Games

Tile Type: square divided in 4 coloured triangles.
Infinite stock available
No rotations or reflections allowed



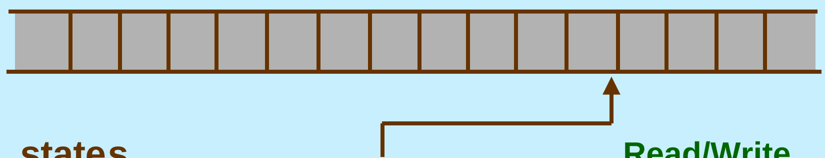
Tiling: Covering of region of the plane such that adjacent tiles have matching colours



Boundary condition: colours given along (part of) edge of region, or some given tile at some given position.

Turing Machine

Tape



Q: states
Σ: tape symbols

Finite Control
Program : P

Read/Write Head

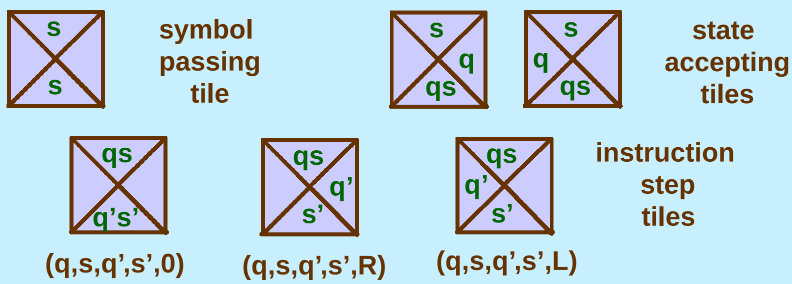
$P \subseteq (Q \times \Sigma) \times (Q \times \Sigma \times \{L,0,R\})$:
(q,s,q',s',m) $\in$ P denotes the instruction:
when reading s in state q print s' perform move m and go to state q'. Nondterminism!

Computations

Configuration c : finite string in $\Sigma^*(Q\times\Sigma)\Sigma^*$
Computation Step c --> c' obtained by
performing an instruction in P
Computation: sequence of steps
Final Configuration: no instruction applicable
Initial Configuration: start state & leftmost
symbol scanned
Complete Computation: computation starting
in initial configuration and terminating
in finite one
Accepting / Rejecting computation ....

Turing Machines and Tilings

Idea: tile a region and let successive color sequences along rows correspond to successive configurations.....



SNAG: Pairs of phantom heads appearing out of nowhere...
Solution: Right and Left Moving States....

Example Turing Machine

K = {q,r,_}
S = {0,1,B}
P = { (q,0,q,0,R),
(q,1,q,1,R),
(q,B,r,B,L),
(r,0,_,1,0),
(r,1,r,0,L),
(r,B,_,1,0) }

q0	1	0	1	1	B
0	q1	0	1	1	B
0	1	q0	1	1	B
0	1	0	q1	1	B
0	1	0	1	q1	B
0	1	0	1	1	qB
0	1	0	1	r1	B
0	1	0	r1	0	B
0	1	r0	0	0	B
0	1	1	0	0	B

Successor Machine;
adds 1 to a binary integer.
_ denotes empty halt state.

11 + 1 = 12

Reduction to Tilings

q0	1	0	1	1	B
0	q1	0	1	1	B
0	1	q0	1	1	B
0	1	0	q1	1	B
0	1	0	1	q1	B
0	1	0	1	1	qB
0	1	0	1	r1	B
0	1	0	r1	0	B
0	1	r0	0	0	B
0	1	1	0	0	B

© Peter van Emde Boas ; 19921029

Peter van Emde Boas: The Games of Computer Science, April 2000

Implementation in Hardware

© Peter van Emde Boas ; 19950310

© Peter van Emde Boas ; 19950310

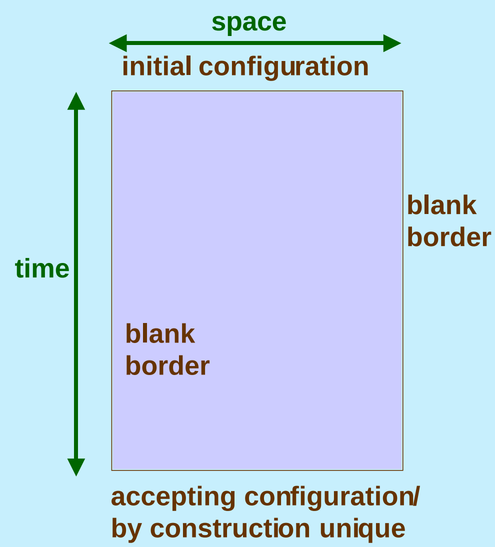
© Peter van Emde Boas ; 19921031

Peter van Emde Boas: The Games of Computer Science, April 2000

Tiling reductions

Program : Tile Types
Input: Boundary condition

Space: Width region
Time: Height region



Tiling Problems

Square Tiling: Tiling a given square with boundary condition: Complete for NP.

Corridor Tiling: Tiling a rectangle with boundary conditions on entrance and exit (length is undetermined): Complete for PSPACE .

Origin Constrained Tiling: Tiling the entire plane with a given Tile at the Origin. Complete for co-RE hence Undecidable

Tiling: Tiling the entire plain without constraints. Still Complete for co-RE (Wang/Berger's Theorem). Hard to Prove!

THE HOLY QUADRINITY

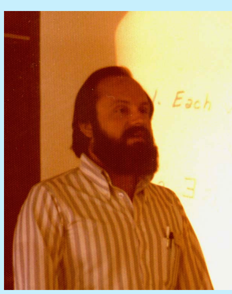
FINITE COMBINATORIAL GAMES

PSPACE

QUANTIFIED PROPOSITIONAL LOGIC: ALTERNATION

UNRESTRICTED UNIFORM PARALLELISM

Walter Savitch



© Peter van Emde Boas

© Peter van Emde Boas

© Peter van Emde Boas

ICSOR; CWI, Aug 1976

San Diego, Oct 1983

UNDERSTANDING PSPACE

- The most Robust Complexity Class
- Solitaire Problem: finding an Accepting path in an Exponentially large, but highly Regular Graph
- Matrix Powering Algorithm: Parallelism
- Recursive Procedure: Savitch Theorem
- Logic: QBF, Alternation, Games

Parallel Computation Thesis

// PTIME = // NPTIME = PSPACE

True for Computational Models which combine Exponential Growth potential with Uniform Behavior.

The **Second Machine Class**

Polynomial Space Configuration Graph

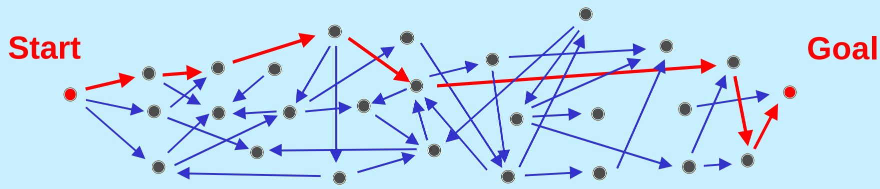
- **Configurations & Transitions:**
  - (finite) State, Focus of Interaction & Memory Contents
  - Transitions are Local (involving State and Memory locations in Focus only; Focus may shift). Only a Finite number of Transitions in a Configuration
  - Input Space doesn´t count for Space Measure

Polynomial Space Configuration Graph

- **Exponential Size Configuration Graph:**
  - input lenght: $|x| = k$; Space bound: $S(k)$
  - Number of States: q (constant)
  - Number of Focus Locations: $k.S(k)^t$ (where t denotes the number of “heads”)
  - Number of Memory Contents: $C^{S(k)}$
  - Together: $q.k.S(k)^t. C^{S(k)} = 2^{O(S(k))}$ (assuming $S(k) = \Omega(\log(k))$)

Polynomial Space Configuration Graph

- **Uniqueness Initial & Final Accepting Configuration:**
  - **Before Accepting Erase Everything**
  - **Return Focus to Starting Positions**
  - **Halt in Unique Accepting State**

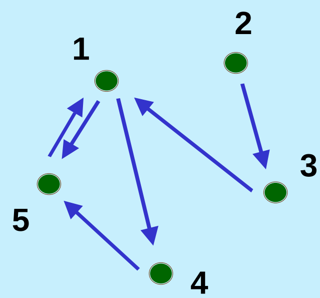


Unreasonable Algorithm

- **Step 1: generate Exponentially large structure**
- **Step 2: Perform Exponentially long heavy computation on this structure**
- **Step 3: Extract a single bit of information from the result - the rest of the efforts are wasted.**
- **:** Ακουε Παντων, Εκλεγε δε ‘α συμφερεισ
- **And this is just what the Parallel Models do.....**

Adjacency Matrix

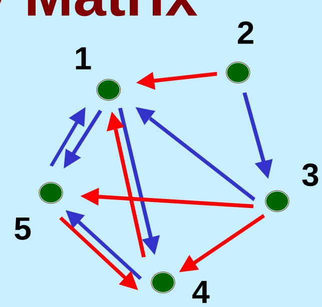
$$M := \begin{bmatrix} 1 & 0 & 0 & 1 & 1 \\ 0 & 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 \\ 1 & 0 & 0 & 0 & 1 \end{bmatrix}$$



Matrix describes Presence of
Edges in Graph;
1 on diagonal: length zero paths

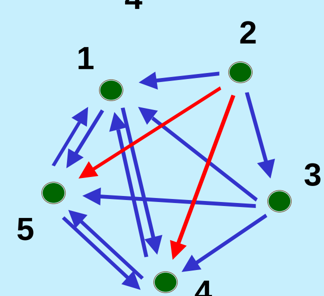
Adjacency Matrix

$$M^2 = \begin{bmatrix} 1 & 0 & 0 & 1 & 1 \\ 1 & 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 1 & 1 \\ 1 & 0 & 0 & 1 & 1 \\ 1 & 0 & 0 & 1 & 1 \end{bmatrix}$$



In Boolean Matrix Algebra
M^2 : Paths up to length 2
M^4 : paths up to length 4

$$M^4 = \begin{bmatrix} 1 & 0 & 0 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \\ 1 & 0 & 1 & 1 & 1 \\ 1 & 0 & 0 & 1 & 1 \\ 1 & 0 & 0 & 1 & 1 \end{bmatrix}$$



Matrix Squaring

$$M[i,j] := \bigvee_{k} (M[i,k] \wedge M[k,j])$$

On an N node graph, a single squaring requires $O(N^3)$ operations

$Log(N)$ squarings are required to compute N-th Power of the Matrix

Remember that $N = 2^{O(S)}$

Think Parallel

- $O(N^3)$ processors can compute these squarings in time
  - $O(\log(N))$ if unbounded fan-in is allowed
  - $O(\log(N)^2)$ if fan-in is bounded
- This is the basis for recognizing PSPACE in polynomial time on PRAM models
- See Second Machine Class paper and/or chapter in Handbook of TCS

Recursive Matrix Squaring

$$M[i,j,p+1] := \bigvee_k (M[i,k,p] \wedge M[k,j,p])$$

M[i,j,0] is the given Adjacency Matrix

Log(N) recursion depth is required to compute N-th Power of the Matrix
Log(N) recursion depth is required to replace N fold Iteration by Recursion
Overall Recursion depth: $\text{Log}(N)^2$

Recursive Path Finding

Reach(x,y,0) := $x = y \vee \text{trans}(x,y)$

Cook/Levin Formula

Reach(x,y,p+1) :=
$\exists z$ [Reach(x,z,p) $\wedge$ Reach(z,y,p)]

Reach(x,y,p+1) :=
$\exists z\, [\forall u,v\, [\ (u=x \wedge v=z) \vee (u=z \wedge v=y) \implies$ Reach(u,v,p)]]

Rabin, Meyer & Stockmeyer trick! The Exponential Growth of the formula is prevented! Size $\approx$ O(S) . O(Rec-depth)

Peter van Emde Boas: The Games of Computer Science, April 2000

Recursive Path Finding

- Space Consumption of Recursive Procedure: O(| stackframe |.depth)
- In this case: | stackframe | and depth are both O(S)
- For path finding Nondeterminism of the original machine is irrelevant!
- Savitch Theorem:
  PSPACE = NPSPACE

Quantified Boolean Formulas (QBF)

INSTANCE: Formula of the form:
F = Qx[Qy[Qz[….. [P(x,y,z, ……)]]] …]
Where P is propositional and Q is ∃ or ∀

QUESTION: is F true ?

THEOREM: QBF is PSPACE-Complete

QBF is PSPACE Complete

Reach(x,y,0) := x = y $\vee$ trans(x,y)

Cook/Levin Formula

Reach(x,y,p+1) :=
$\exists$z [Reach(x,z,p) $\wedge$ Reach(z,y,p)]

Reach(x,y,p+1) :=
$\exists$z [$\forall$u,v [(u=x $\wedge$v=z) $\vee$(u=z $\wedge$v=y) ==>
Reach(u,v,p)]]

The resulting formula is polynomial size and reduces an arbitrary PSPACE computation to QBF .
Brute force Evaluation runs in PSPACE.

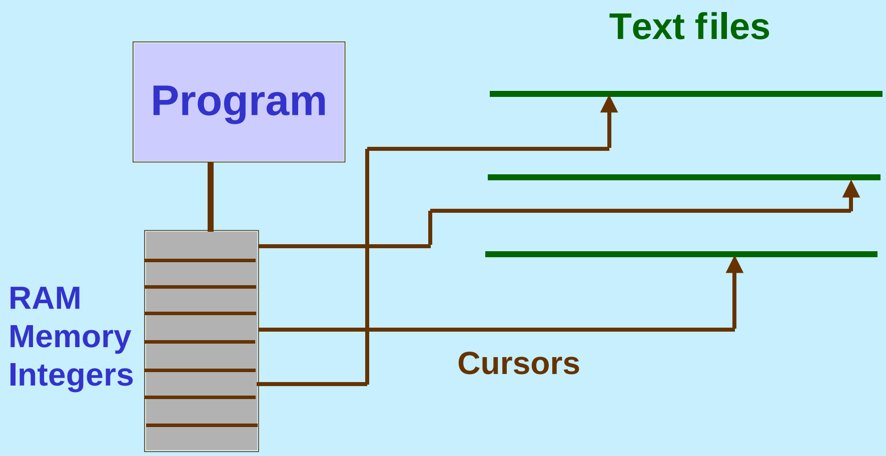
The Power of your Editor

A Simple model of a Text Editor (EDITRAM) solves QBF in polynomial time.

Similar sequential models with the power of parallelism:
Vector Machines (Pratt, Rabin & Stockmeyer 74)
MRAM (Hartmanis & Simon 74, Bertoni et. al 81)
ASSM (Tromp & vEB 93)
Vector Machines (Iwama & Iwamoto 98)

So have patience when your word processor makes you wait .....

EDITRAM



EDITRAM Instructions

The standard RAM instructions on the main memory (store, load, store-I, load-I, ...)

Read at cursor position
Write at cursor position
Move cursor (one position, to address, to end,..)
Copy segment of text (marked by pair of cursors)
Systematic string replacement (literal strings only) : C / aab / aacba /

EDITRAM Program for QBF

1. Eliminate Quantifiers and variables:
   Starting at innermost quantifier replace
   $\forall x_i$ F(...x_i ...) by (F(... 0 ....) $\wedge$ F(... 1 ...))
   $\exists$ x_i F(...x_i ...) by (F(... 0 ....) $\vee$ F(... 1 ...))
2. Eliminate connectives
   Working inside out evaluate logical connectives: C / (0 $\vee$ 1) / 1 / etc.
3. The result is a literal 0 or 1 .
   Read the answer .

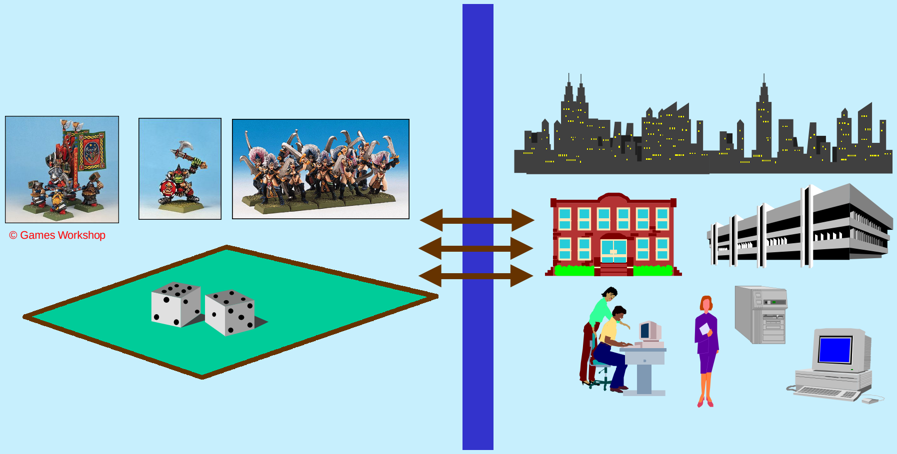
step 1 requires a subroutine for identifying and marking variables (due to literal only condition)

Similar new models

ASSM: A variant of a pointer machine (Kolmogorov, Uspenski, Schönhage) which accesses and generate nodes in parallel by allowing for reversed edges in paths....

Vector Machine (I & I): transforms vectors in square matrices by row (column) replication with corresponding converse folding operations. (far more restricted compared to original VM)

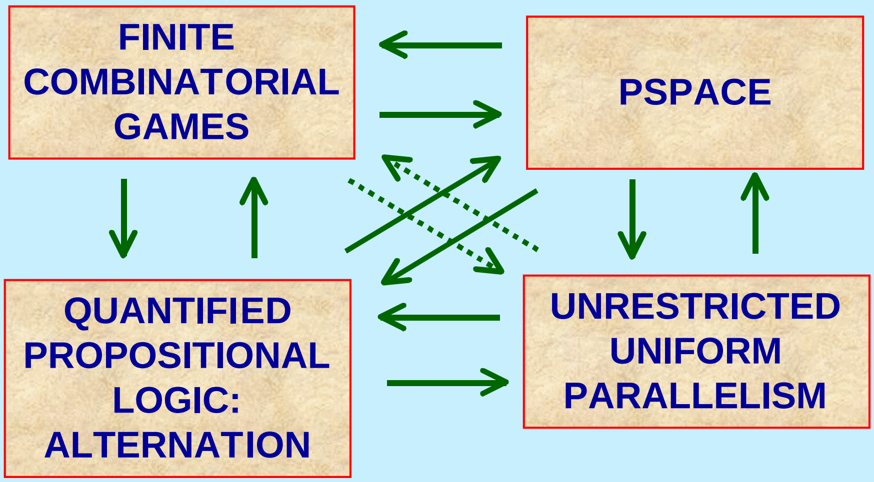
Part II, Connecting Games and Computer Science



Topics

- Computation, Games and Computer Science
- Recognizing Languages by Games; Games as Acceptors
- Understanding the connection with PSPACE (The Holy Quadrinity)
- Interactive Protocols and Games
- Loose Ends in the Model ?

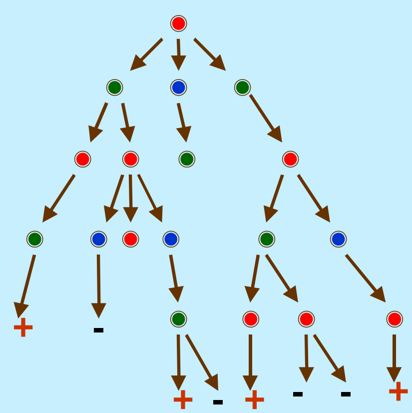
THE HOLY QUADRINITY



Alternating Computation

Configuration Type

- ● Existential
- ● Universal
- ● Negating
- + Accepting
- - Rejecting



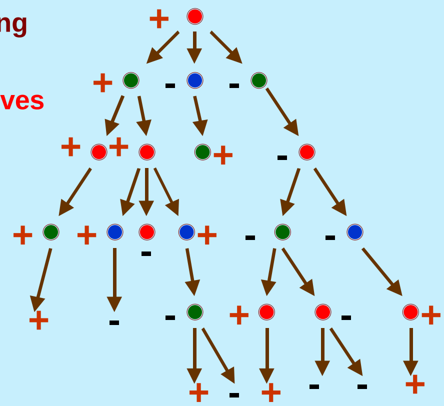
Computation Tree

Peter van Emde Boas: The Games of Computer Science, April 2000

Alternating Computation

Configuration Type: Game meaning

- Existential: Thorgrim moves
- Universal: Urgat moves
- Negating: Role Switch
- \+ Accepting: win (for Thorgrim)
- \- Rejecting: Loose



Evaluation Full Computation Tree

This Tree Accepts

This is just backward induction on a game tree ;

But what is the Game ??

Alternating Computation as a Game

Negating states are unnecessary - by dualizing parts of the computation tree they can be removed.

Infinite branches don't contribute to the game value (non-trivial to prove)

What remains is a Computation Game where both Thorgrim and Urgat control nondeterministic choices in the computation. Thorgrim wants the computation to accept. Urgat wants to prevent this from happening......

QBF as a LOGIC GAME

- Game is a Propositional Formula $\Phi(x_1,x_2, \ldots.\ x_n,\ y_1,y_2, \ldots.\ y_n)$
- THORGRIM and URGAT select values for the x_i and y_i in a specified order
- THORGRIM wins if the formula eventually evaluates to true: otherwise URGAT wins the game.
- Easy to solve on an ATM
- Cook-Levin Reduction from ATM to QBF shows that QBF is ATM-P hard.

The Alternation Theorems

THM 1: APTIME $\subseteq$ PSPACE:

Recursive evaluation of the quality of the root of the Alternating Computation Tree:
Depth = O(time) , | Stackframe | = O(time)

Resulting Overhead: Square

The Alternation Theorems

THM 2: PSPACE $\subseteq$ APTIME:

QBF trivially can be solved by an ATM
QBF is PSPACE-hard
What Further Evidence do we Need?

Resulting Overhead: Polynomial: $O(n^4)$

The reduction to QBF squares the space and the number of variables. Aside from substituting 0 or 1 for these variables the QBF evaluation takes linear time…

Exercise !

The Alternation Theorems

THM 3: ALOGSPACE $\subseteq$ PTIME:

Iterative evaluation of the quality of the all nodes in the Alternating Computation Tree:

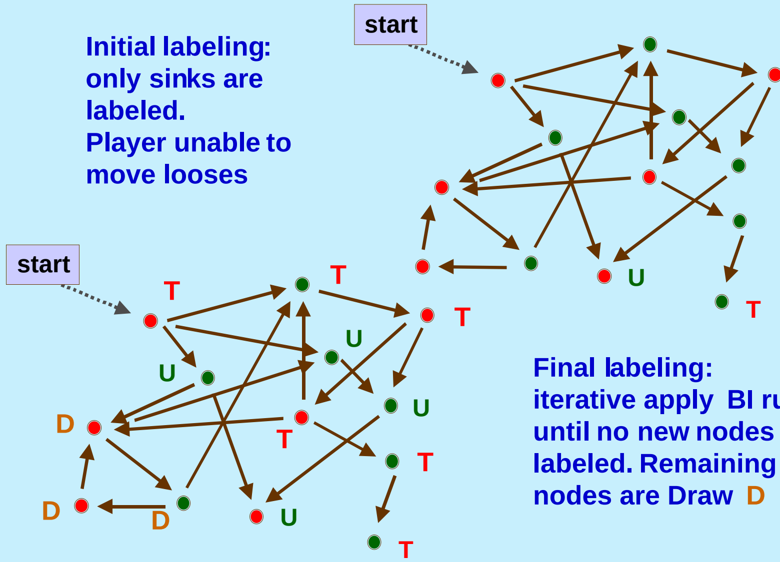
#Nodes = Exponential in S; # Stages = O(# Nodes)

In terms of Games: This is Backward Induction on a Game Graph.

Peter van Emde Boas: The Games of Computer Science, April 2000

Backward induction on Game Graphs

Initial labeling:
only sinks are
labeled.
Player unable to
move looses



Final labeling:
iterative apply BI rules
until no new nodes are
labeled. Remaining
nodes are Draw D

The Alternation Theorems

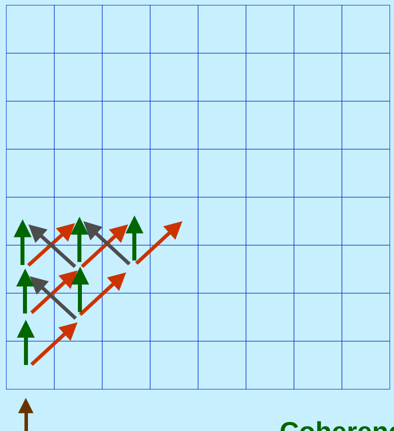
THM 4: PTIME $\subseteq$ ALOGSPACE:

Guess a Space-Time diagram of Accepting
Computation, (thinking in terms of a correct tiling)
starting from the “accepting” tile and moving
Backwards in time:

At cell X guess contents of three upper neighbors;
Universally validate these three upper neighbors.

THIS IS CORRECT DUE TO DETERMINISM!

Guess and Validate



“accepting Configuration”

Coherence of validated Guesses is enforced from Top to bottom

Two Player Tiling Game

1	2	3	4	5	6	7	8
9	10	11					

Thorgrim and Urgat fill successive rows from left to right in order;
Thorgrim fills the odd columns, and Urgat the even ones.
Board width is even. Thorgrim wins if legal tiling is constructed;
otherwise Urgat wins the game.

Two Person Tiling Game

If the Tiling represents a Turing Machine Computation Constructing a Legal Tiling, respecting the Border Condition, corresponds to an Accepting Alternating Computation.
Thorgrim wins by simulating a winning strategy on the Alternating Turing Machine (understood as game).
Urgat, however has additional possibilities: he may destroy the legal encoding of a machine computation

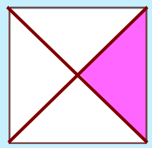
Urgat therefore must be forced to stay within the constraints allowed by the encoding.

Conventions on ATM

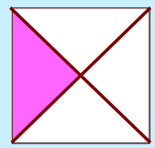
- Tape uses even number of cells
- Universal and Existential States Alternate
- Left or Right moving States only
- All Instructions Move
- Unique Accepting Configuration

Chlebus´ Bag of Tricks

Thorgrim and Urgat, each obtain their own set of tile types; this is enforced by introducing two flavors of the vertical colors, indicated by a pink shade.



Base Shading for Thorgrim

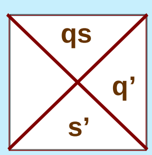


Base Shading for Urgat

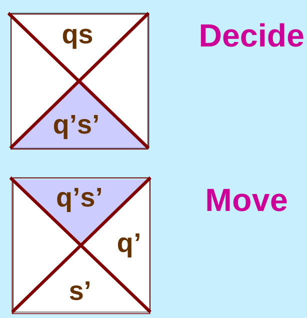
Both borders of Rectangle to be tiled are shaded white.

Chlebus´ Bag of Tricks

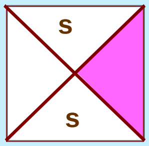
Tiles types representing the Turing Machine instructions are replaced by pairs: Decide & Move



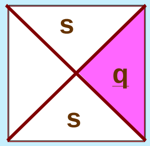
The blue shading indicates a right moving state q’
for left moving states use grey shading



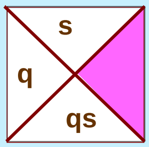
THORGRIM'S TILES



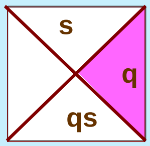
pass symbol & prevent intro from right



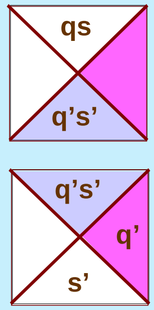
pass symbol & force intro from right for Urgat



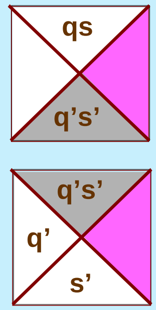
accept from left



accept from right

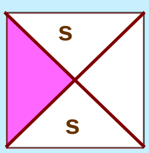


Decide & move right

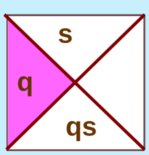


Decide & move left

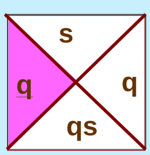
URGAT'S TILES



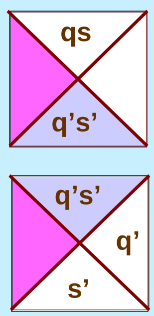
pass symbol; accept from right is prevented



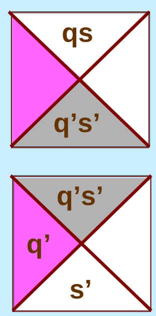
accept from left



accept from right is forced



Decide & move right



Decide & move left

Known Examples of Games used in Complexity Theory (1980+)

- Tiling Games (NP, PSPACE, NEXPTIME,....)
- Pebbling Game (PSPACE)
- Geography (PSPACE)
- HEX (generalized or pure) (PSPACE)
- Checkers, Go (PSPACE)
- Block Moving Problems (PSPACE)
- Chess (EXPTIME)

The Common View is that Games Characterize PSPACE

Types of Games (and Computations)

- Single player - no choices : Routine : Determinism
- Single player - choices : Solitaire : Nondeterminism
- Two players – choices : Finite Combinatorial Games : Alternating Computation
- Single player - random moves : Gambling : Probabilistic Algorithms
- Two players - choices and random moves : Interactive Proof Systems
- Several players - concurrent moves : Multi Prover Systems

Peter van Emde Boas: The Games of Computer Science, April 2000

One vs Two Sided Error

On input x a random computation is performed with probability p(x) to accept. Purpose: determine membership in some language L.

One sided error: for some $\varepsilon > 0$ it holds that:
$x \in L \Rightarrow p(x) > \varepsilon$ $\quad$ $x \notin L \Rightarrow p(x) = 0$

Two sided bounded error: for some $\varepsilon > 1/2$ it holds that:
$x \in L \Rightarrow p(x) > \varepsilon$ $\quad$ $x \notin L \Rightarrow p(x) < 1-\varepsilon$

Two sided unbounded error: it holds that:
$x \in L \Rightarrow p(x) > 1/2$ $\quad$ $x \notin L \Rightarrow p(x) < 1/2$

Randomized Complexity Classes

RP: class of languages recognized by P-time one sided error devices

BPP: class of languages recognized by P-time two sided bounded error devices

PP: class of languages recognized by P-time two sided unbounded error devices

Amplification for One Sided Error

The computation on input x is repeated k times:

$x \in L$ =>
probability of at least one succes $> 1 - (1-\varepsilon)^k$

$x \notin L$ => probability of at least one succes = 0

So the chance of false judgement can be made arbitrarily small. In fact: for all $c > 0$ smaller than $O(|x|^{-c})$, provided the initial error rate $\varepsilon > |x|^{-c'}$ for some $c' > 0$.

Amplification for Two Sided Error

Given a Random Event X with possible outcomes 0 and 1 with a probability of succeeding $p \in [0,1]$.

If $p \neq 1/2$ the chance of one outcome is greater than that of the other. We want another event where the chance of the more frequent outcome is even larger, in fact as close to 1 as one likes.

Idea: make k independent observations, and select the majority outcome....

Question: how large should k be in order to reach chance $> 1 - \delta$ for a given $\delta > 0$?

Amplification Lemma

Answer: it suffices to select k = O(| log(δ) |)

Proof: let $\gamma := \varepsilon(1-\varepsilon)$, then $\gamma < 1/4$
WLOG: $\varepsilon > 1/2$ so 1 is more probable.
The probability that the majority event is 0 is bounded by:

$$\sum_{j=0}^{k/2}\binom{k}{j}\left(\varepsilon^{j}(1-\varepsilon)^{(k-j)}\right) \le \sum_{j=0}^{k/2}\binom{k}{j}\left(\varepsilon^{k/2}(1-\varepsilon)^{(k/2)}\right) =$$

$$= \gamma^{k/2}\sum_{j=0}^{k/2}\binom{k}{j} \le \gamma^{k/2}2^{k} = (4\gamma)^{k/2}$$

Use of Amplification

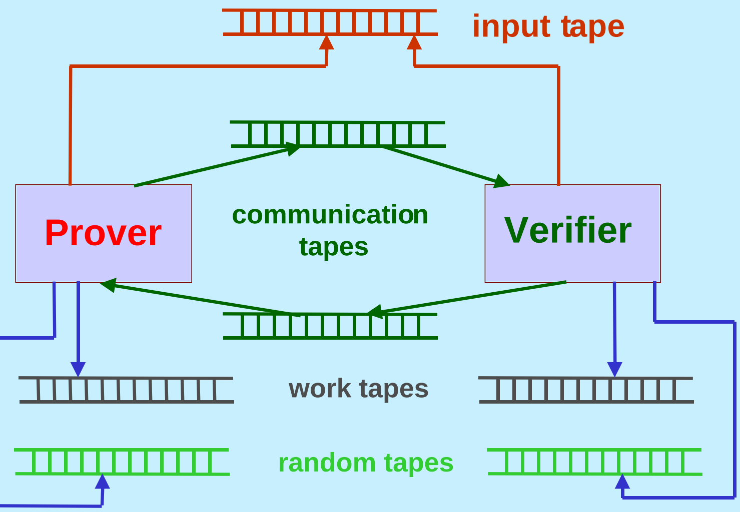
When ε is unequal from 1/2 the Majority Experiment decides the more probable outcome with an error rate decreasing Exponentially in the number of trials.

The implicit constant depends on how far ε is bounded away from 1/2 .

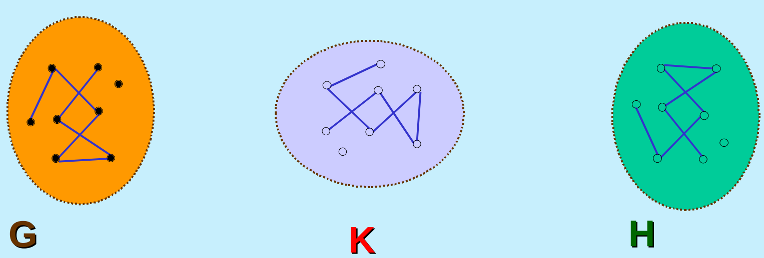
The majority experiment can be performed both sequentially and in parallel.

If ε = 1/2 Amplification is not possible. So errors in the two sided bounded error model can be made arbitrarily small, but in the unbounded error model performing more trials achieves nothing....

The Basic Interactive Model



Example: Graph Isomorphism

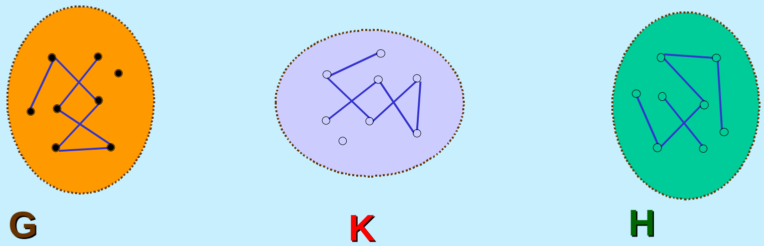


Claim: G and H are Isomorph

Prover: submits K obtained by permuting either G or H
Verifier: asks at random to show K ≈ G or K ≈ H
Prover: provides required isomorphism
Failure Probability (in case non-isomorph graphs) = 1/2

ZERO KNOWLEDGE !!!

Example: Graph Non-Isomorphism



Claim: G and H are Non-Isomorph

Verifier: Submits K obtained by permuting either G or H
Prover: Tells whether K results from G or from H
Failure Probability (in case non-isomorph graphs) = 1/2

ZERO KNOWLEDGE !!!

Using the Interactive Model

One of P or V opens the Communication

Next both Participants Exchange a Sequence of Messages, based on:

- Contents Private Memory
- Input
- Visible Coin Flips
- Earlier Messages (Send and) Received so far
- Current Message

At some point V decides to Accept the input (I am convinced - you win) or to Reject it (I don't Believe you - you loose)

Computational Assumptions

- Verifier is a P-time bounded Probabilistic Device
- Prover (in principle) can do everything (restrictions => feasibility)
- All messages and the number of messages are P-bounded.

Accepting a Language L

- For every x in L the Prover P has a Strategy which with High Probability will convince the Verifier
- For every x outside L , regardless the strategy followed by the Prover, the Verifier will reject with High Probability

IP = class of languages accepted by Interactive Proof Systems = PSPACE

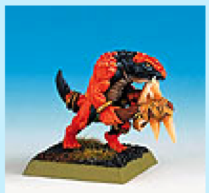
The Participants

© the Games Workshop

Thorgrim; our wise Prover

© the Games Workshop

Urgat; our skeptical Verifier



© the Games Workshop

Stragtos; fully deterministic

© the Games Workshop

Orion; Random moves only

Various Models

Verifier vs. Prover

Stragtos vs. Orion: Probabilistic Computation
Rabin, Strassen Solovay

Orion vs. Thorgrim: Games against Nature
unbounded error Papadimitriou’s model

Orion vs. Thorgrim: Arthur Merlin Games
Babai & Moran

Urgat vs. Thorgrim: Interactive Protocols
Goldwasser Micali Rackoff

Where is the Beef ?

The name of the area: Interactive Protocols, suggests that Interaction is the newly added ingredient.

Interaction already resides in the Alternating Computation Model!

The Key Addition therefore is Randomization.

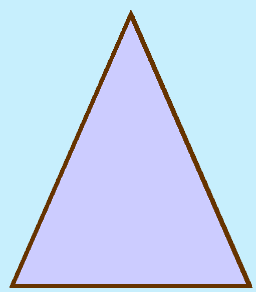
Leaf Languages

Nondeterministic Computation Tree with Ordered Binary Choices Everywhere.

Yields string of 2^T labels at leafs.

Accepts on the basis of some property of this string.

Backward Induction only for Regular properties (but where is the Game??)



++-+---+-+---++----.....--+-+

Can Leaf Languages be analyzed by Games?

GEOGRAPHY

Objects Selected: Directed Edges

Constraint: Edge is connected to the previously selected edge

Winning: Player unable to move looses

Morale: Both players build a maximal Eulerian Path.

GEOGRAPHY is PSPACE HARD

Proof: Reduction from QBF

Special Contraints on QBF:
Propositional Formula in CNF

$$Qx_1\ Qx_2\ ....\ Qx_k\ [\ C_1 \wedge C_2 \wedge\ ...\ \wedge C_m\ ]$$

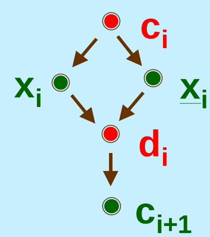
$Q = \forall$ or $\exists$, C_j are clauses

Peter van Emde Boas: The Games of Computer Science, April 2000

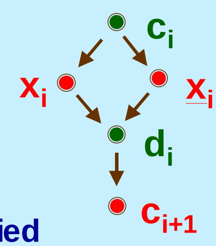
GEOGRAPHY is PSPACE HARD

Component Design: Vars

selection component
for $\exists\ x_i$ followed
by $\forall\ x_{i+1}$



selection component
for $\forall\ x_i$ followed
by $\exists\ x_{i+1}$

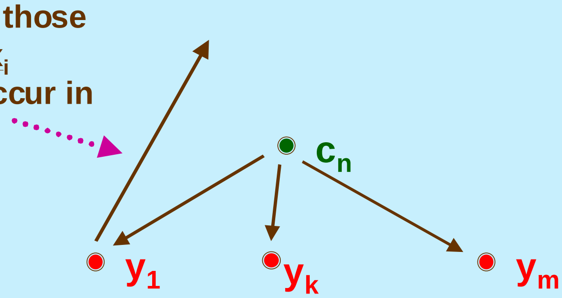


if no alternation occurs the
nodes d_i and c_{i+1} are identified

GEOGRAPHY is PSPACE HARD

Component Design: Clauses

towards those
x_i and $\underline{x}_i$
which occur in
C_1



Here it is crucial that Urgat chooses for a clause and Thorgrim chooses a literal in the clause

SCHAEFER´s Variations

- Positive formulas only (in CNF)
- Thorgrim selects always true, Urgat selects only false
- Explicit Ordering Relaxed or Removed
- Shared Variables are Possible
- Both players flip a variable to true; the last player to move wins (looses)

GEOGRAPHY is PSPACE HARD

c_1 is the startnode

First Player depends on the Type of First Quantifier: if it is Existential, Thorgrim will start the game as usual.

Play: each play will first traverse the vars components and select a truth assignement. Subsequently Urgat selects a clause, Thorgrim one of its literals. Only if this literal node is unvisited in the first stage Urgat has a move left.

STRATEGY MAPPING

Players select truth values for their variables as in the QBF logic game.

If the formula now is false Urgat can select it; and all literals chosen by Thorgrim will be unvisited, and Urgat still has a free move to a visited out-degree 1 node. Urgat wins.

If the formula now is true Urgat must select a true clause, so Thorgrim can select a visited literal. Urgat can't go anywhere. Thorgrim wins. NB! Urgat is to move, even if at this node Thorgrim had to move previously!

NODE KAYLES

Objects Selected: Nodes

Constraint: Node is not connected to any previously selected node

Winning: Player unable to move looses

Morale: Both players build a maximal Independent Set.

NODE KAYLES is PSPACE HARD

Proof: Reduction from QBF

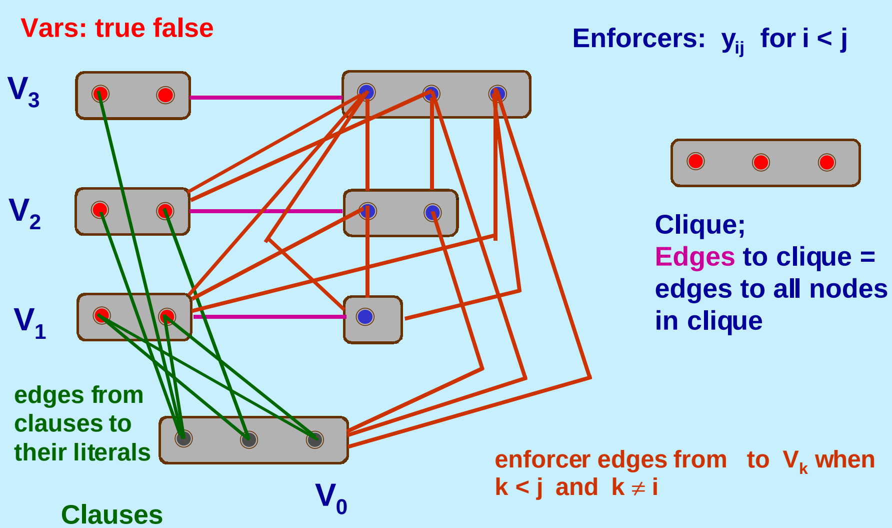
Special Contraints on QBF:

- Quantifiers alternate
- First and last quantifier are Existential
- Propositional Formula in CNF
- $C_1 = x_1 \vee \underline{x}_1$

$Qx_n\ Qx_{n-1} \ldots\ Qx_1\ [\ C_1 \wedge C_2 \wedge \ \ldots \wedge C_m\ ]$

$Q = \forall$ or $\exists$, C_j are clauses

NODE KAYLES is PSPACE HARD



NODE KAYLES is PSPACE HARD

Regular Play: Players select truth value in order.
Next Urgat must select a clause node. This requires that none of its literal nodes has been selected,
so the formula should evaluate to false and Urgat has selected the false clause.

Deviant Play: a Player which selects any node which is legal but violates the protocol of order is punished by an immediate loss:
Selection vars node in V_k with $k < i$ ==> y_{ki} is lethal
Selection of enforcer y_{ki} ==> x_k is lethal.