CONCLUSION

Tilings code Turingmachine computations conveniently

Using tilings in reductions saves bothering about head positions, states and tape symbols

Resulting reductions come out to be elegant (sometimes improving on the most elegant ones available in the literature; E.G., Hilbert 10 example)

Taking the computational origin into consideration is sometimes necessary; E.G., PDL example)

A BINARY COUNTER IN PDL

Additional propositional variables:

$c_0,\ldots,c_L$ Bits of a binary counter , $L = O(n)$

Initially all counter bits equal 1:

$$c_0 \wedge \ldots \wedge c_L$$

When c_L is decremented the game lasts too long:

$$[a^*]\,(\, z \Leftrightarrow \neg\, c_L\,)$$

Every move the counter is decremented:

$$[a^*]\,(\,(\,\neg\, c_0 \wedge \neg\, c_1 \wedge \ldots \wedge \neg\, c_{j-1} \wedge c_j\,) \Rightarrow$$

$$(\quad [a]\,(\, c_0 \wedge c_1 \wedge \ldots \wedge c_{j-1} \wedge \neg\, c_j\,) \wedge$$

$$(\,(\, c_{j+1} \Rightarrow [a]\, c_{j+1}\,) \wedge (\,\neg\, c_{j+1} \Rightarrow [a]\,\neg\, c_{j+1}\,)\,) \wedge$$

$$\ldots\ldots \wedge$$

$$(\,(\, c_{L+1} \Rightarrow [a]\, c_{L+1}\,) \wedge (\,\neg\, c_{L+1} \Rightarrow [a]\,\neg\, c_{L+1}\,)\,)$$

$$)\,)$$

(for $j = 0, \ldots, L$)

This later formula also proves that a polynomial size PDL formula can describe a model with exponentially long terminating branches.

THE REDUCTION TO PDL

No placement at two positions at the same time:

$[a^*] (\neg q_i \vee \neg q_{i'}) \quad (i \neq i')$

and placements proceed in the corect order:

$[a^*] ((q_1 \Rightarrow [a]q_2) \wedge .... \wedge (q_n \Rightarrow [a]q_1))$

Everywhere a tile must be placed:

$[a^*] (p_{i,u} \vee p_{i,w} \vee p_{i,1} \vee \cdots \vee p_{i,n})$

but no two tiles at the same place:

$[a^*] (\neg p_{i,j} \vee \neg p_{i,j'}) \quad (j \neq j')$

The adjacency conditions must be observed:

$[a^*] ((q_i \wedge p_{i,t} \wedge p_{i-1,t'}) \Rightarrow [a] \neg p_{i,t''}), (t,t',t'') \in \mathrm{INC}$

but, when Alice moves, all legal moves may be chosen:

$[a^*]((\neg b \wedge q_i \wedge p_{i,t} \wedge p_{i-1,t'}) \Rightarrow \langle a \rangle p_{i,t''}),(t,t',t'') \notin \mathrm{INC}$

Either Elias wins, or the game continues, unless it proceeds for too many moves:

$[a^*] (p1,w \vee (\langle a \rangle(b \vee \neg b) \wedge \neg z) \vee z)$

Remains to ensure that z indeed expresses that the game takes too long:

THE REDUCTION TO PDL

The reduction uses a single program a , representing a move in the game.

The propositional variables used:

$p_{i,j}$	**Tile type j is placed in column i (i= 0,..,n+1)**
b	**It is Elias' turn to move**
q_i	**Next move in column i**
z	**The game has taken too long**

Assume that the regular winning situation for Elias is signaled by placing tile w in column 1.

Tile type u represents a white tile enforcing the boundary conditions.

Initial Configuration:

$$b \wedge q_1 \wedge p_{0,u} \wedge p_{1,t1} \wedge \cdots \wedge p_{n,tn} \wedge p_{n+1,u}$$

Alternation of moves:

$$[a^*] ((b \Rightarrow [a] \neg b) \wedge (\neg b \Rightarrow [a] b))$$

Somewhere a tile must be placed:

$$[a^*] (q_1 \vee q_2 \vee .... \vee q_n)$$

CHLEBUS to the RESCUE

Alternating Tiling of a Coridor:

Input: A set of tile types, and a pair of equal sized colored walls *start* and *finish*. Elias and Alice in turn place a tile observing the tiling adjacency constraints, producing a coridor with white side-walls connecting start and finish. Elias wins if this construction succeeds; otherwise Alice wins.

Question: Does Elias have a winning strategy?

In this form an elegant reduction to PDL satisfiability can be given.

SNAG: In this way nonterminating games become a problem.

Fisher & Ladner evade this problem by presuming that the Alternating Turing machines used for their reduction always terminate;

This assumption has no reasonable equivalent in terms of tiling games.

Solution: Declare the game won by Elias when it drags on for more than $2.n.s^n$ moves
where $n = |start| = |finish|$ and $s = \#$ Tile types

CHLEBUS to the RESCUE

Direct translation of ATM computations into Tiling games, using the coding tiles used before yields completeness for Tiling games where each player in turn tiles *an entire row*.

Tiling a Coridor with Alternating Rows:

Input: A set of tile types, and a pair of equal sized colored walls *start* and *finish*. Elias and Alice in turn place a row of tiles observing the tiling adjacency constraints, producing a coridor with white side-walls connecting start and finish. Elias wins if this construction succeeds; otherwise Alice wins.

Question: Does Elias have a winning strategy?

The moves in these games are too complex to support an elegant reduction to PDL satisfiability.

Bogdan Chlebus shows that the more reasonable convention where both players in turn place a single tile leads to the same completeness results.

This result is obtained by engineering tiles so that Elias forces Alice to stay within the realm of properly encoded TM computations without being able to deviate from this coding himself

APPLICATION TO PDL

The Satisfiability problems for formulas in *Propositional Dynamic Logic* is known to be complete for EXP. Lowerbound by Fischer & Ladner from 1979; Upperbound by Pratt 1979.

Proof based on reduction of PSPACE Alternating Turing Machines, using APSPACE = EXP.

w is accepted by ATM M iff

a suitable formula $\Phi(w,M)$ in PDL has a model.

By the reduction this model then must contain a subtree of the full computation tree of M on w witnessing the fact that M accepts w.

The standard connection between accepting Alternating Turing machines and winning strategies in 2-person games:

w is accepted by ATM M iff

The initial configuration corresponding to w represents a winning position for player 1 in some 2-person game corresponding to M where player 1 controls the Existential and player□2 the Universal moves.

APPLICATION TO PDL

PDL Propositional Dynamic Logic: a logic for reasoning about the behavior of regular programs on an abstract interpretation.

Language : **two sorts: propositions and programs**

Proposition vars: p,q,.... **metavars:** ϕ,ϕ'

Program vars: a,b,.... **metavars:** π,π'

Syntax :

$$f ::= p, \phi \vee \phi, \phi \wedge \phi, \neg \phi, [\pi]\phi, <\pi>\phi,$$

$$\pi ::= a, \pi; \pi, \pi \cup \pi, \pi^*, \phi?$$

Semantics : **Kripke frame: F := (K,V,R)**

Set of states K; state vars: x,y,...

p: predicate: subset V_p of K

a: atomic program: relation R_a in K X K

Satisfaction: $F,x \models \phi$

combined with interpretation R_π for all programs π

EG: $F,x \models [\pi]\phi$ **iff** $F,y \models \phi$ **for all** $y \in R_\pi(x)$

$$R_{\phi?} = \{<x,x> \mid F,x \models \phi\}$$

EXAMPLE [Jones, 1982]

$x \in W_{<z,u,y>}$ **iff** he following system is solvable over **N** :

$eLg^2+a' = (b-xy)q^2$, $e = y + mt$, $n = q^{16}$,

$q = b^{5^{60}}$, $L' + q^4 = 1 + L'b^5$, $t' + 2z = b^5$, $L = u + tt'$,

$$r = [\, g+eq^3+Lq^5 + (2(e-zL')(1+xb^5+g)^4 + L'b^5q^4)q^4](n^2-n) \\ + [q^3 -bL + L + t'L'q^3 + (b^5-2)q^5][n^2-1] \;,$$

$p = 2ws^2r^2n^2$, $p^2k^2- k^2+1 = T'^2$, $4(c-ksn^2)^2 + h' = k^2$,

$k = r + 1 + hp - h$, $a = (wn^2+1)rsn^2$, $c = 2r+1+f'$,

$d = bw + ca - 2c - 4a'c' -5c'$, $d^2 = (a^2 -1)c^2 + 1$,

$f^2 = (a^2 -1)i^2c^4 + 1$,

$(d+of)^2 = ((a+f^2(d^2-a))^2 -1)(2r+1+jc)^2 + 1$.

Final System

Proposition: for every Turing Machine A one can effectively compute numbers r (a power of 2), K, p, q,

x_k, y_k, x'_k, y'_k, for $0 \le k \le K$, $x_{a'}$, $x'_{a'}$, $y_{b'}$, $y'_{b'}$, $x'_{a''}$,

such that the Turing machine A halts on a blanco input tape

iff

the following exponential diophantine system of equations is solvable in the unknowns

S, T, W_0, ..., W_K, Z_0, ... , Z_K, Z", R, U, B, E :

$Z_0W_0 + ... + Z_KW_K + B + E = (r^{(T+1)(S+1)(p+q)} -1)/(r-1)$

-- $Z_0 + ... + Z_K = Z''$ *%% everywhere a tile*

-- Z_0 **<u>dom</u>** Z'' , ... , Z_K **<u>dom</u>** Z'' *%% at most one tile*

-- $R = r^{(p+q)}$, $U = R^{S+1}$ *%% yardsticks*

-- $Z'' = (R^{(T+1)(S+1)} -1)/(R-1)$ *%% all square positions*

-- $w_0 = (x_0 + y_0 + x'_0U + y'_0R)$, ... , *%% tile values*

$w_K = (x_K + y_K + x'_KU + y'_KR)$

-- $B = ((R^{S+1} -1)/(R-1))(x'_{a'} + U^T x_{a'})$ + *%% horizontal border*

$((U^{T+1} -1)/(U-1))(y'_{b'} + R^S y_{b'})$ + *%% vertical border*

$(x'_{a''} - x'_{a'})$ *%% correction initial state*

-- $E = ((U^{T+1} -1)/(U-1))R^S(r^p -1)/(r-1)$ + *%% border excess*

$((R^{S+1} -1)/(R-1))\, U^T\, r^p\, (r^q -1)/(r-1)$

The KNAPSACK-like equation

Σ value(T_j) . value(locations of T_j) + value boundary =

= total value entire grid within square

The condition that everywhere some tile is located and that no two tiles cover the same location are expressed by one more linear equation and (for each tile type) an invocation of the Kummer relation:

Σ value(placements of T_j) =

value all possible placements

value(locations of T_j) <u>dom</u> value all possible placements

The definition of these values and of the boundary values requires two more exponential yardsticks

$R := r^{(p+q)}$, $U = R^{S+1}$,

B : total value of boundary constraint points on edge of square,

E : total value of excess gridpoints added for balancing horizontal and vertical edges

Totals and boundary extra's are computed by summing geometric series like:

Total = $\Sigma\ r^j$, $j = 0 ,\ldots , (S+1).(T+1).(p+q)-1$

WHY THE TILE-PLACEMENT TERMS FACTORIZE

Use a systematic numbering of the points along the grid edges:

p = # horizontal colours

q = # vertical colours

x_i, y_i, x'_i, y'_i : the (tile-type dependent values representing the choice of a color singleton or co-singleton set of points along an edge)

Base position for the square at position row i, column j
$L = (i.S + j)(p+q)$, contributing value r^L

moving a square right :
position $L+u$ => position $L + (p+q) + u$

moving a square down :
position $L + v$ => position $L + (S+1)(p+q) + v$

The value of placing tile T_j at position <i,j> :

$$r^{(p+q)[i.S+j]} . \{x_j + y_j + x'_j . r^{(p+q)(S+1)} + y'_j . r^{(p+q)}\}$$

position-value ***Tile-value*** **W_j**

The reduction from the halting problem

$M_i(X)$ halts iff

for the tile types corresponding to M_i and the left-top border condition corresponding to input X there exists a tiling of a finite square in the plane with white right and bottom boundary.

NEXT: use technique from previous chain of reductions
BOUNDED TILING ≤ EXACT COVER ≤ KNAPSACK

where however all placements of a single tile type are represented by a single term.

This is possible since the Knapsack value representing a single tile placement factorizes as a product:

i) a factor depending on the position only and

ii) a factor depending only on the tile type.

So placings of the same tile type can be added.

THE KUMMER PREDICATE

X <u>dom</u> Y : The binary representations of X and Y

<u>bin</u> X and <u>bin</u> Y satisfy:

<u>bin</u> X has a 1 only where <u>bin</u> Y has a 1

Examples: 3 <u>dom</u> 7 , since 011 <u>dom</u> 111

not 5 <u>dom</u> 6 , since not 101 <u>dom</u> 110

Kummer 1852: X <u>dom</u> Y iff the binomial coefficient $\binom{Y}{X}$ is odd

More general result due to Lucas 1878:

if $Y = Y_n \ldots Y_0$ (radix p) and

$X = X_n \ldots X_0$ (radix p) , then

$$\binom{Y}{X} = \binom{Y_n}{X_n} \cdots \binom{Y_0}{X_0} \ \underline{\mathrm{mod}}\ p \quad , \ p \text{ prime}$$

Proof: Evaluate $(1 + T)^Y$ in $Z/(pZ)[T]$ and compare coefficients of T^X.

Definition Binomial coefficients: $M = \binom{N}{K}$ iff

$U = 2^{N+1}$ and $M < U$ and $V < U^K$

and $(U + 1)^N = W.U^{K+1} + M.U^K + V$

(only binary exponentials used !)

(compare the powers of the decimal number 1001)

EXAMPLES OF DIOPHANTINE RELATIONS

X is a square:	$X = Y^2$
X is nonprime:	$X = (Y + 2)(Z + 2)$
$X < T$	$T = X + Y + 1$
$X \mid T$	$T = X.Y$
$X \underline{\bmod} T = U$	$X = T.Y + U$, **and**
	$T = U + Z + 1$

Closure under logical operations:

$P = 0$ **and** $Q = 0$ $\Leftrightarrow$ $P^2 + Q^2 = 0$

$P = 0$ **or** $Q = 0$ $\Leftrightarrow$ $P.Q = 0$

Existential quantification is within the language: no problem.

Problems arise out of the operations :negation and Universal quantification

The hunt for the global complexity:

Find universal equations with few unknowns and of low degree:

Result [Jones '82] The pairs (# unknowns, degree) listed below are universal:

(58,4)	**(21,96)**
(38,8)	**(19,2668)**
(32,12)	**(14, ± 200000)**
(29,16)	**(13, 6.6 E 43)**
(28,20)	**(12, 1.3 E 44)**
(26,24)	**(11, 4.6 E 44)**
(25,28)	**(10, 8.6 E 44)**
(24,36)	**(9, 1.6 E 45)**

Local complexity: estimate the size of the solutions:
Adleman & Manders 1975 - 1977 : Diophantine complexity and diophantine machine model

HILBERT 10 proof requires to establish that the diophantine language is sufficiently powerful to talk about machine computations.
David Putnam, Robinson: half-way step, where also exponentials are allowed like $X = 2^Y$.
Matijasevich (1970) eliminates exponentials.

THE HILBERT 10 PROBLEM

Input: A diophantine Equation

$Q(X_1, \ldots, X_n) = 0$

Question: Is this equation soilvable over the natural numbers (or the integers) ?

HILBERT 10: There exists no algorithm deciding this question.

Related issues:

(W_i is the i-th RE set in some standard enumeration)

Diophantine definability:

x in W_i <=> $P_i(x, Y_1, \ldots, Y_n) = 0$ is solvable

Universal polynomial:

x in W_i <=> $Q(i, x, Y_1, \ldots, Y_n) = 0$ is solvable

x in W_i <=> $x = R_i(Y_1, \ldots, Y_n) \geq 0$ is solvable

From EXACT COVER to KNAPSACK

Use the standard reduction from Karp.

Let V be the set of numbers $\{0,1,....,s\}$

For F_j take the number $f_j := \sum_{x \in F_j} r^x$ 0010011....10 , radix r

For the target number take $v := \sum_{x \in V} r^x$ 1111111....11 , radix r

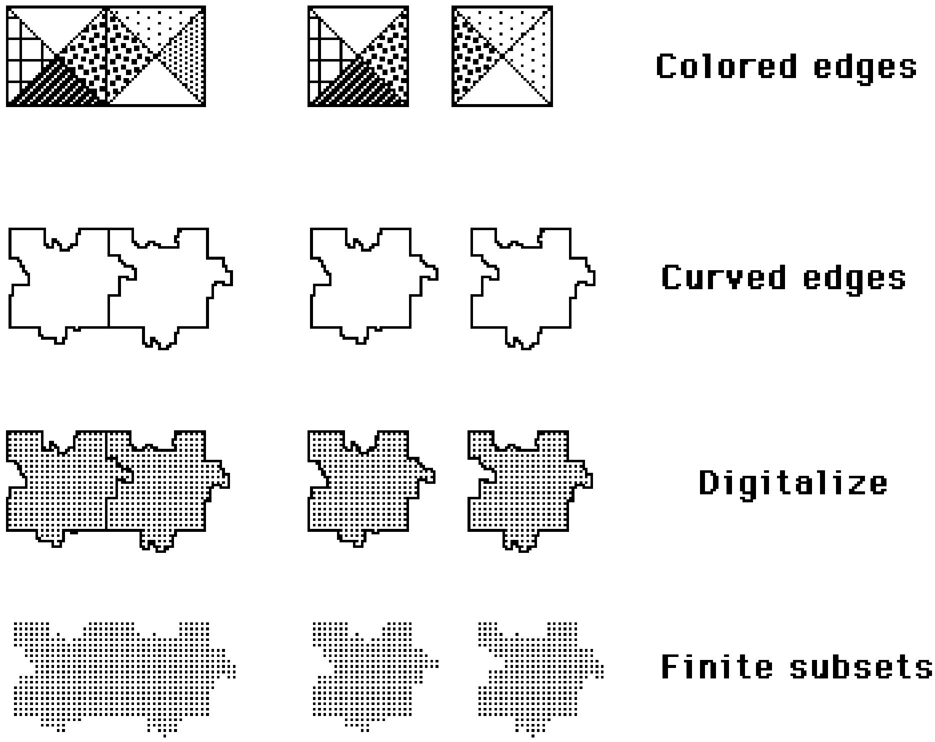
We can represent v by a sum of f_j iff the EXACT COVER instance is solvable.

Required is $r >$ m (the number of elements in F).

Preferably take $r = 2^s$ for a suitable s .

Note that each number f_i still represents placing a tile, except for the number representing the boundary condition.

THE REDUCTION IN PICTURES



For the reduction to work only the grid points along the edges matter.

The bumps and notches include or exclude a single gridpoint along the edge corresponding to the edge color.

Digitalization, restricted to gridpoints along edges

THE REDUCTION FROM TILINGS TO SET COVER

SET COVER: Given, a finite set V and a collection of subsets $\mathcal{F} = (F_i)_{i=1,\ldots,m}$. Does there exist a subcollection $\mathcal{F}' \subseteq \mathcal{F}$ for which V becomes the disjoint union of $\mathcal{F}'$.

Idea behind the reduction:

Replace in the given BOUNDED TILING instance the colored edges by curved edges, like in a real-life Jigg-saw puzzle.
Replace these pieces by their digitalized versions. Solving the puzzle now amounts to covering the points in the puzzle region, which is digitalized as well.
The boundary condition is enforced by a single piece which must be chosen in order to obtain a covering at all.

Inspection shows that the digitalized pieces can be reduced to singleton or co-singleton sets representing the possible colours along the edge connecting two adjacent squares

THE REDUCTION TO CLIQUE

The basic idea equals the reduction SAT $\leq$ CLIQUE

Finding a clique amounts to finding a tiling

Graph:

Nodes: Gridpoints X Tile-types + {*border* }

Edges: (p_1,t_1) and (p_2,t_2) are connected when:

* $p_1 \neq p_2$ **are not adjacent positions or**

** p_1 **and** p_2 **are adjacent positions and the colours of** t_1 **and** t_2 **match along the common edge**

(p,t) and *border* are connected if p is not adjacent to the border , or
p is adjacent to the border and t obeys the applicable border constraint

Now a clique of size # Gridpoints + 1 represents a solution of the tiling problem and conversily

Reductions to other problems

From bounded tiling to SATISFIABILITY:

For given tile types $T = \{T_1,...,T_t\}$ and a given $n \times n$ square , we introduce propositional variables p_{ijk}, with $1 \le i,j \le n$, $1 \le k \le t$, denoting:

a tile of type T_k *is located on position* $[i,j]$.

The tiling conditions now are expressed by simple propositional formulas.

On every location there is a tile:

$(p_{ij1} \vee p_{ij2} \vee .... \vee p_{ijt})$

No two tiles on a single position:

$(\neg p_{ij1} \vee \neg p_{ij2}) \wedge (\neg p_{ij1} \vee \neg p_{ij3}) \wedge ... \wedge (\neg p_{ijt-1} \vee \neg p_{ijt})$

The tiles T_k and T_m that don't match should not be placed adjacent to each other:

$(\neg p_{11k} \vee \neg p_{21m}) \wedge (\neg p_{21k} \vee \neg p_{31m}) \wedge .. \wedge (\neg p_{n-1nk} \vee \neg p_{nnm})$

and similar for vertical pairs

When tile T_k doesn't fit the border condition on location $[1,j]$ don't put it there: $\neg p_{1jk}$

and similar for other boundary conditions.

The resulting formula is the conjunction of these subformulas; its size is $O(n^2)$.

OTHER FACTS ABOUT TILINGS

Undecidability: The undecidability result for planar tiling doesn't require the placement of the initial tile at the origin. The *unconstrained tiling problem* is unsolvable as well. The proof becomes quite complicated
(Berger, Robinson, and (more recently) B. Durand)

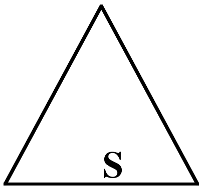
Periodic tiling: There exist tiles sets for which a planar tiling exists but no periodic tilings (tilings obtained by lifting a tiling of the torus to the plane).

Tiling games: Two person full information games where players place tiles alternatively and where the game is lost by the first player unable to move, with an suitable winning condition for completed games.
Determination of the player having a winning strategy becomes another decision problem. (Chlebus, 1986)

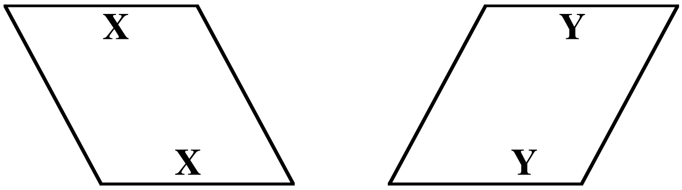
Bounded tiling for two players turns out to be **PSPACE-complete**.
Bounded rectangle tiling for two players is EXP-complete.

Adjacency conditions: the matching colour condition can be generalized to arbitrary horizontal and vertical adjacency conditions (EG. Levin 1973); this can be easily reduced to the matching colour condition.

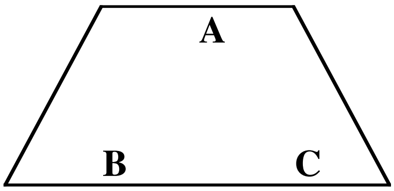
Tile Types for grammar G



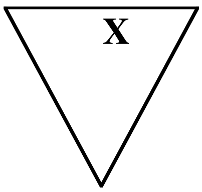
Tile for start symbol S



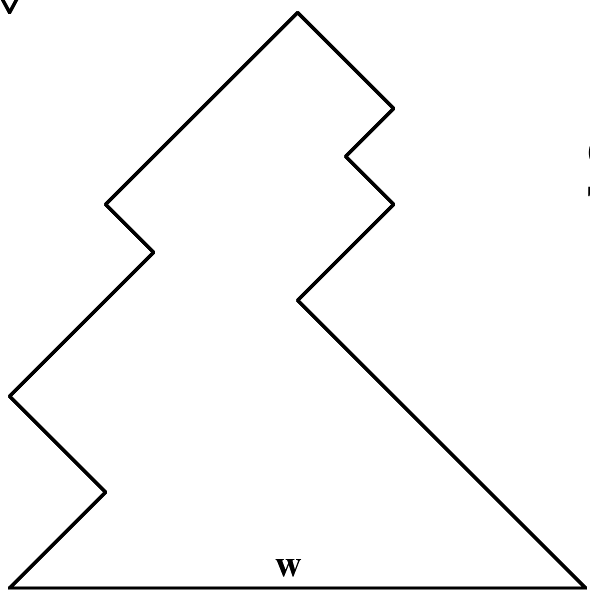
Filler Tiles



Tile for production A --> BC



Tile for production X --> ε



Coridor to be Tiled

A P-complete Tiling Problem

(with Cheat)

P-complete problems are problems in P believed to be *inherently sequential* since they are hard for P under LOGSPACE (or NC^1) reductions.

Known examples include:

Circuit Value problem

Linear (in)equalities over the rationals

Generated set over semigroup

Context-free parsing with ε-rules

The later problem is easily encoded using tiles of the shapes indicated on the next slide:

CF Grammar $G = (N,T,P,S)$; input $w \in T^*$

Now $w \in L(G)$ iff there exists a tiling of a coridor with blank "crooked" walls connecting a point entrance with a horizontal exit wall colored by x .

Examples of tiling problems

Bounded tiling: For a given coloring on the boundary tile a give square ? : ***Complete for*** **NP**

Coridor tiling: For a given pair of colored walls, is it possible to tile a given coridor with white walls connecting the two walls; the length of the coridor is unconstrained: **PSPACE-*complete*.**

Origin constrained tiling: Someone has placed a tile in the origin of the plane; extend it to a tiling of the complete plane: ***Complete for*** Π^0_1 **(*I.E., Undecidable*)**

Periodic tiling: Given some particular tile; does there exist a tiling of the plane where this tile is used infinitely often?

Complete for Σ^1_1 **(I.E., Highly intractible)**

(used by Harel for proving highly undecidability for a variety of programming logics and other intensional logics)

QUESTION: Describe an interesting tiling problem which is **P-complete** ?

A Hierarchy of unsolvable problems

Given a set of tile types T we may ask for the solvability of the tiling problem for various kinds of boundary conditions.

The resulting problems turn out to be complete for various complexity classes characterized by the Turing machine model.

P Polynomial time

NP Nondeterministic Polynomial Time

PSPACE Deterministic Polynomial space

(= Nondeterministic Polynomial Space)

EXP Deterministic Exponential Time

Π^0_1 Undecidable

Σ^1_1 Outside arithemetical hierarchy

Completenss for some class amounts to representing

the hardest problem within the class .

Completeness is established using *reductions:*

(as illustrated in the master reduction above)

f: A => B is a reduction provided:

* f is easy to compute (Ptime or even Logspace)

** x in A iff f(x) in B : $A = f^{-1}(B)$

The Master Reduction

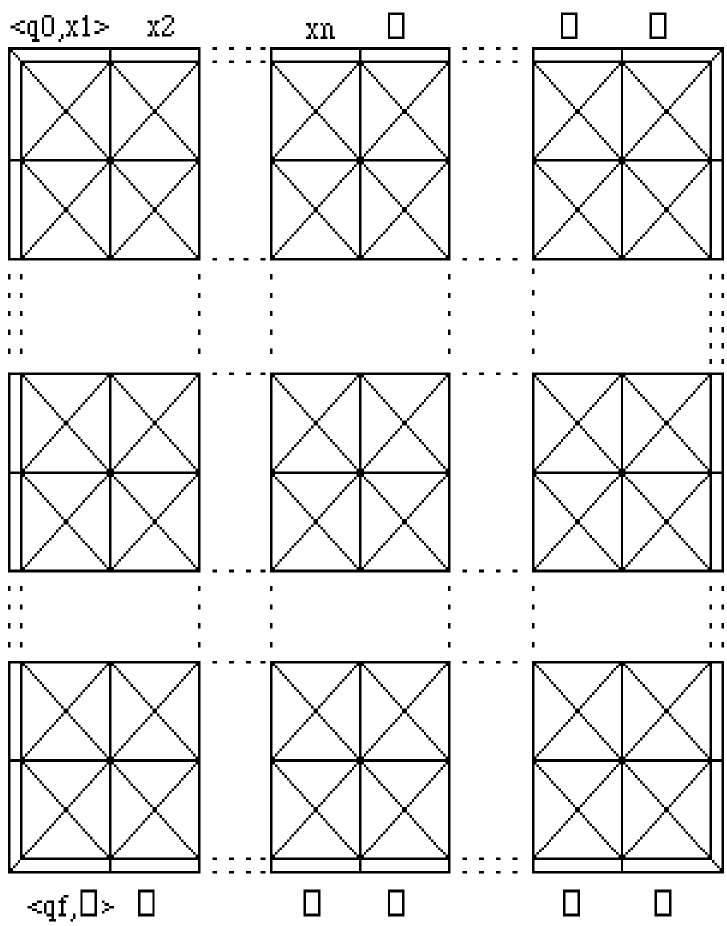
Let A be a problem in **NP**, then there exists a Nondeterministic Turing machine M and a exponent k such that A is accepted by M in time $k.n^k$.

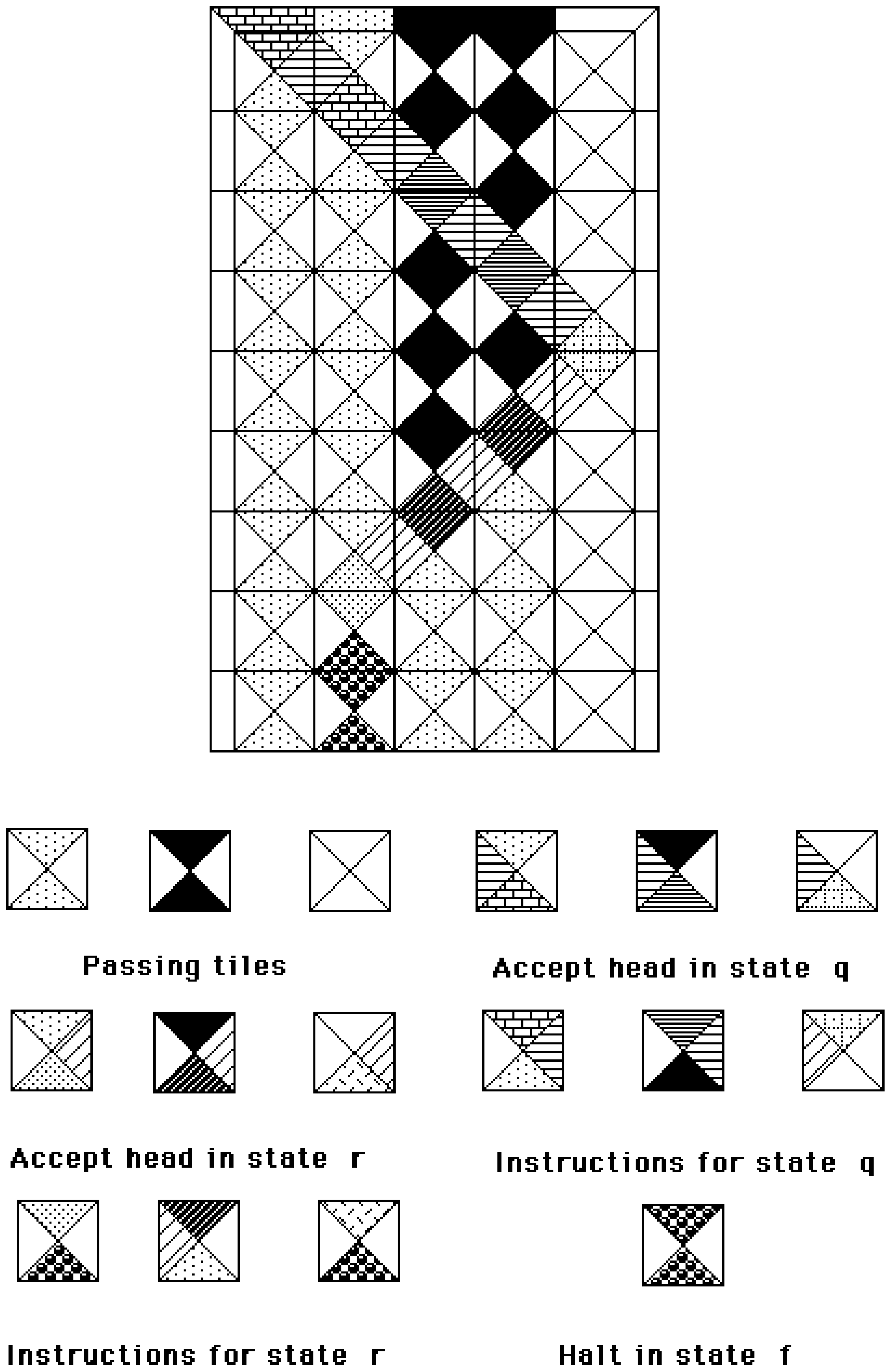
Without loss of generality M accepts in a unique halting state, after having erased all contents on the tape.

Let $T = \{T_1, \ldots , T_m\}$ be the set of tile types corresponding to the program of M. and consider some input x; then we have:

$x \in A$ **iff** x is recognized by M in time $K := k.(|x|)^k$ **iff**

The square of size K X K below can be tiled with tiles in T:





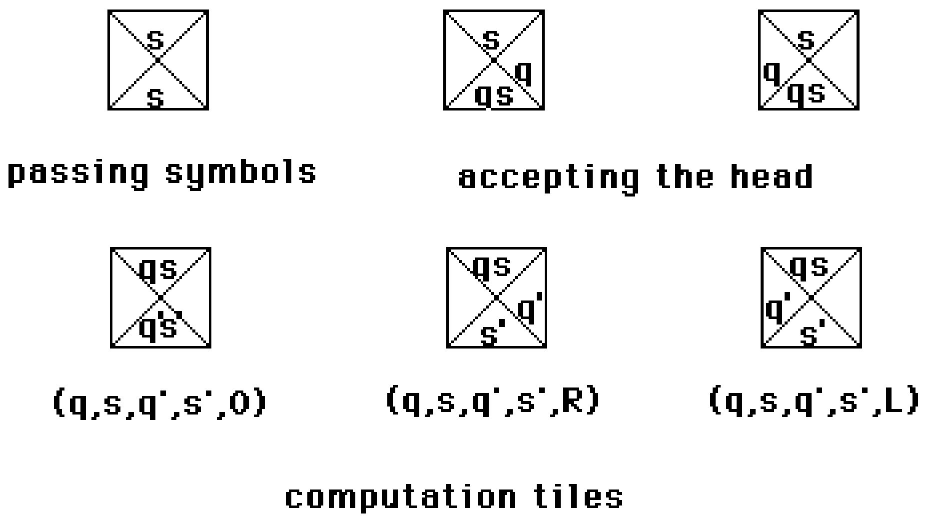
Computation of the successor machine encoded as a tiling instance

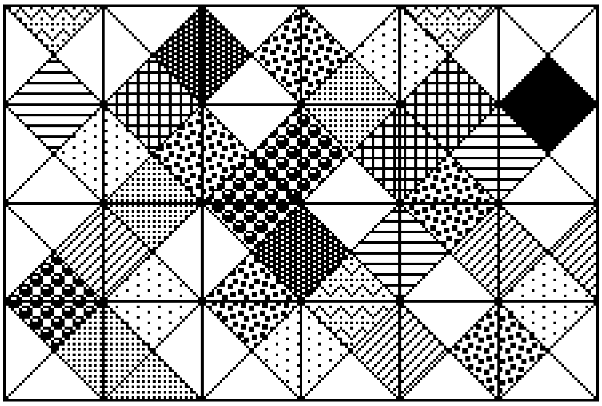
Reduction of Turing machine computations to tiling problems

For a given Turingmachine program, which, WLOG satisfies some technical conditions, a set of tile-types can be constructed which will simulate the computations of this machine.

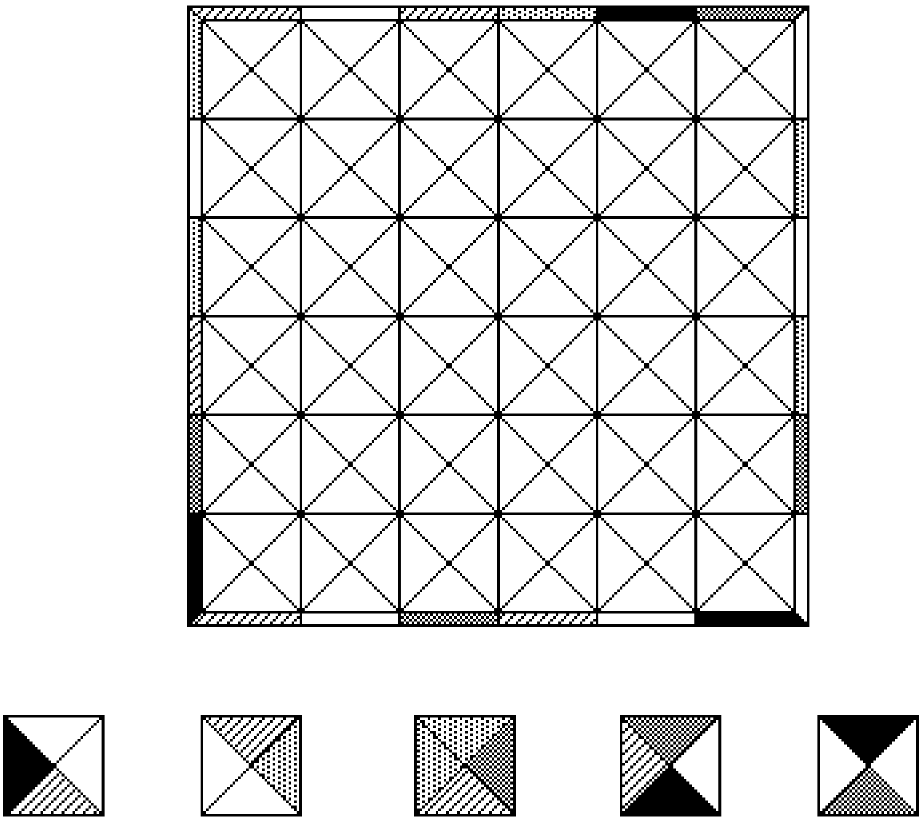
The boundary conditions will ensure that the computation operates on the required input and that time and/or space constraints are enforced.

Solvability of the resulting instance of BOUNDED TILING entails that the intended computation terminates and accepts the input within the given time and/or space bound.





Example of a legal fragment of a tiled region



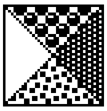
Instance of BOUNDED TILING

THE TILING GAME

Instance: A collection $T = \{T_1,\ldots,T_S\}$ of *tile-types* and a two-dimensional region to be tiled.

Question: Is it possible to tile the given region, using the given set of tile-types ?

Tile-type: Square with four colored edges



Boundary condition: A coloring on the edge of the region; alternatively, a single tile placed somewhere.

Further rules

Adjacent tiles must share colours along their common edge

Tiles adjacent to the boundary should conform to the colours given along the boundary edges

For every available tile-type an unrestricted amount of tiles is available; the tiles may not be rotated or reflected

THE SUCCESSOR MACHINE

M given by the following program :

K = {q,r,f} ; q_0 = q ; F = { f } ;

S = { 0,1,❐ } ;

P = { (q,0,q,0,R) , (q,1,q,1,R) , (q,❐,r,❐,L),
(r,0,f,1,0) , (r,1,r,0,L) , (r,❐,f,1,0) }

A computation: Time-Space diagram

¢	[q0]	0	1	1	❐	$	|---
¢	0	[q0]	1	1	❐	$	|---
¢	0	0	[q1]	1	❐	$	|---
¢	0	0	1	[q1]	❐	$	|---
¢	0	0	1	1	[q ❐]	$	|---
¢	0	0	1	[r1]	❐	$	|---
¢	0	0	[r1]	0	❐	$	|---
¢	0	[r0]	0	0	❐	$	|---
¢	0	[f1]	0	0	❐	$	

FACT: Working out the Cook-Levin reduction for a n-step computation in terms of this diagram yields a SATISFIABILITY instance in DNF of length $O(n^2)$. (which improves the $O(n^3)$ of the standard proof)

Configurations and computations

Configuration: Tape contents, state and head position coded by : $C := ¢\ s_0 s_1 \ldots s_{p-1}\ [q,s]\ s_{p+1} \ldots s_m\ \$$

Computation step: impact of performing a single program instruction: $(q,s,q',s',R)\ \varepsilon\ P$:

$$¢\ s_0 s_1 \ldots s_{p-1}\ [q,s]\ s_{p+1} \ldots s_m\ \$ \qquad \vdash\!\!--$$

$$¢\ s_0 s_1 \ldots s_{p-1}\ s'\ [q' s_{p+1}]\ s_{p+2} \ldots s_m\ \$$$

(missing symbols on the boundary of the tape are filled out by the blank symbol ❒).

Computation: sequence of configurations connected by computation steps.

Initial configuration: $¢\ [q_0,s_1]\ s_2 \ldots s_n\ \$$

on input $s_1 s_2 \ldots s_n$.

Final configuration: $¢\ s_0 s_1 \ldots s_{p-1}\ [qs]\ s_{p+1} \ldots s_m\ \$$,

with P containing no instruction starting with <q, s, ...>.

Full computation :

Computation connecting an initial and a final configuration.

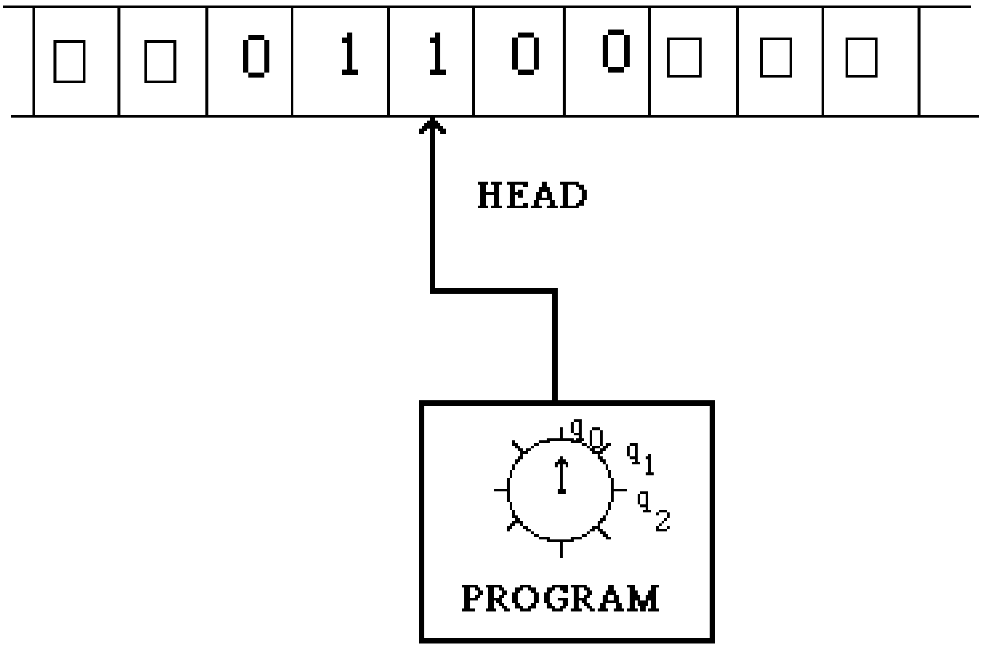
Accepting computation:

Full computation terminating in an accepting state ;

(otherwise computation is called **Rejecting**)

THE TURING MACHINE MODEL

Computational model of A. Turing , 1936. ; currently named **Turingmachine.**



K	*Finite set of states*
q_0	*Initial state in* K
S	*Finite set of tape symbols*
❑	*Blank symbol in* S
{R,0,L}	*Moves*
P	*Program* , a relation
	P: (K x S) => (K x S x {R,0,L})
F	*Set of accepting states* F $\subseteq$ S

(q,s,q',s',m) $\in$ P : *In state q, while reading s, print s', perform move m and transfer to state q'*

Contents

WARNING:

Familiarity with some basic complexity theory and knowledge of propositional dynamic logic is assumed

The Convenience of Tilings

Peter van Emde Boas

ILLC, FWI , Univ. of Amsterdam

31 March 1995