SAT Encodings vs. Dancing Links for Polyomino Tiling

Jan Jungfer

University of Basel

19.11.2025

Contents

Exact Cover

Definition (Cover)

Let X be a finite set and let $S \subseteq \mathcal{P}(X)$. A **cover** of X is a subset $S^* \subseteq S$ such that

$$\bigcup_{E \in S^*} E = X.$$

Definition (Exact Cover)

An **exact cover** of X is a cover $S^* \subseteq S$ such that

$$E_i \cap E_j = \emptyset \quad \text{for all distinct } E_i, E_j \in S^*.$$

Exact Cover

Example:

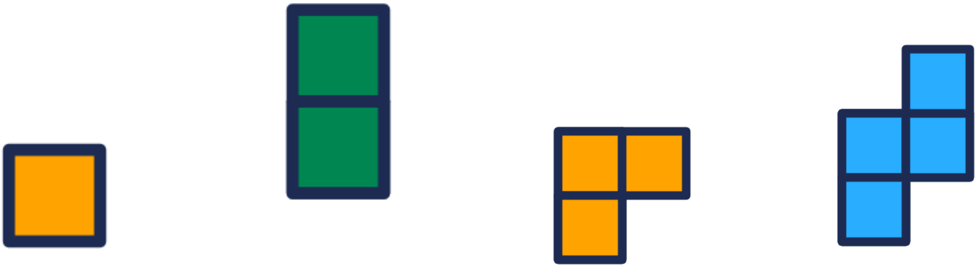
Let $X = \{a, b, c, d\}$ and $S = \{\{a, b\}, \{a, b, c\}, \{c, d\}\}$.

- $\{\{a, b, c\}, \{c, d\}\}$ is a cover of X.
- $\{\{a, b\}, \{c, d\}\}$ is an exact cover of X.

Definition (Exact Cover Problem)

The **exact cover problem** is the problem of finding an exact cover $S^* \subseteq S$ for a given set X and a collection of subsets S.

Polyomino tiling



Monomino Domino Tromino Tetromino

Figure: Polyomioes

Polyomino Tiling

- Free: The polyomino can be rotated and mirrored. All these variations are considered to be the same polyomino.
- One-sided: The polyomino can only be rotated. As above, all rotation are considered to be the same polyomino. The mirrored polyomino is considered to be distinct.
- Fixed: The polyomino can neither be rotated nor mirrored. All variations are considered to be distinct polyominoes.

Polyomino Tiling

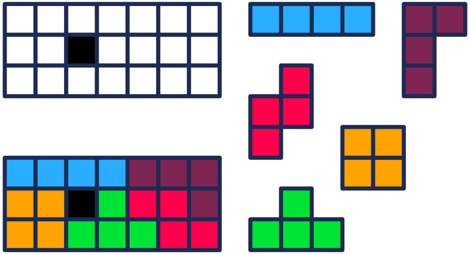
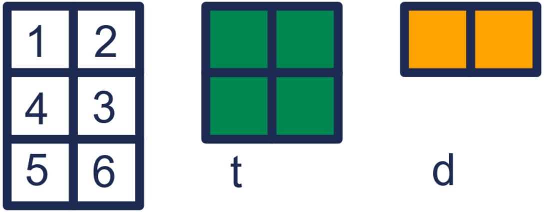


Figure: Tetromino Tiling Problem

Polyomino Tiling



$X = \{d, t, 1, 2, 3, 4, 5, 6\}$
$S = \{\{d, 1, 2\}, \{d, 3, 4\}, \{d, 5, 6\}, \{t, 1, 2, 3, 4\}, \{t, 3, 4, 5, 6\}\}$

Polyomino Tiling

The problem can be encoded in a binary matrix M where each row represents an element s of S and each column represents an element x of X. The element $M_{i,j} = 1$ if $x \in s$ otherwise 0.

Example:

$X = \{d, t, 1, 2, 3, 4, 5, 6\}$

$S = \{\{d, 1, 2\}, \{d, 3, 4\}, \{d, 5, 6\}, \{t, 1, 2, 3, 4\}, \{t, 3, 4, 5, 6\}\}$

$$\begin{pmatrix} 1 & 0 & 1 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 1 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 1 & 1 \\ 0 & 1 & 1 & 1 & 1 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 & 1 & 1 & 1 \end{pmatrix}$$

DLX

Knuth's Algorithm X:

1. Systematically covers columns by selecting a row that covers it.
2. Removes covered columns and incompatible rows from the matrix.
3. Backtracks when next column can't be covered. -> Reintroduces removed columns and rows.
4. Minimises branching by selecting columns with few ones.
5. Explores all partial and full solutions

Dancing Links (DLX):

1. Implemetation of Algorithm X with a four way linked list.
2. Sparse matrix is memory efficient
3. Rows and columns are quickly added and removed by manipulating links.

SAT

Definition (Boolean Formula)

A **Boolean formula** is a formula constructed using Boolean variables, the logical operators $\neg$ (not), $\wedge$ (and), and $\vee$ (or). A Boolean variable can take the value `true` or `false`.

Definition (Assignment)

An **assignment** is a mapping from the set of Boolean variables in a formula to the set $\{\texttt{true}, \texttt{false}\}$. An assignment specifies a truth value for each variable in the formula.

SAT

Definition (Evaluation)

Given a Boolean formula φ and an assignment α, the **evaluation** of φ under α, denoted $\varphi(\alpha)$, is the result of recursively substituting the truth values of variables according to α and applying the logical operators. The evaluation yields either `true` or `false`.

Definition (Conjunctive Normal Form)

A Boolean formula is in **conjunctive normal form** (CNF) if it is a conjunction of clauses, where each clause is a disjunction of literals. A **literal** is either a Boolean variable or its negation.

SAT

Definition (Boolean Satisfiability Problem)

The **Boolean satisfiability problem (SAT)** is the problem of determining whether a given Boolean formula has an assignment of truth values to its variables that makes the formula evaluate to `true`.

Direct Encoding

- Variables: We have one variable for every row
- Constraints: We enforce the exactly-one-constraint for every column
  - Exactly-one-constraint:
    At-least-one constraint: One clause containing all positive literals of the placements that cover the cell/polyomino. For every column j of M:

$$\bigvee_{i : M_{i,j} = 1} x_i$$

2. At-most-one constraint: One clause for every pair of incompatible variables x_a and x_b:

$$\neg x_a \vee \neg x_b$$

Bimander Encoding

- Combination of the Commander and the Binary encodings.
- It replaces the previous at-most-one encoding.
- Incompatible variables are split into m equally sized groups.
- Each group has a unique binary code. Only one code can be active at a time and only the group with the active code can contain a variable that's assigned true.
- For every n incompatible variables we require an additional $\lceil \log_2(m) \rceil$ auxiliary variables but only $n * \lceil \log_2(m) \rceil$ clauses in total.

Bimander Encoding

For group G_h with binary code $C_h \in \{0,1\}^{\lceil \log_2(m) \rceil}$ ($C_h[k]$ being the kth bit of C_h and b_k the corresponding variable) and variable $x_i \in G_h$, the following clauses are added:

$$\bigwedge_{k=0}^{\lceil \log_2(m) \rceil - 1} \begin{cases} (\neg x_i \vee b_k), & \text{if } C_h[k] = 1 \\ (\neg x_i \vee \neg b_k), & \text{if } C_h[k] = 0 \end{cases}$$

Cell-based Encoding

- Specific to polyomino tiling.
- Each cell encodes which polyomino covers it with a binary code.
- For p polyominoes each cell requires $\lceil \log_2(p) \rceil$ additional auxiliary variables
- Neighbouring cells have to be covered by polyominos in a way that they form a valid polyomino.
- At-most-one for polyominoes has to be enforced separately.
- For each polyomino P represented by code C_P and each variable x that is a placement of P covering m cells $c_1, c_2, \ldots c_j \ldots c_m$, we enforce:

$$x \implies \bigwedge_{k=0}^{\lceil \log_2 p \rceil - 1} \bigwedge_{j=1}^{m} \begin{cases} (b_{c_j,k}), & \text{if the } k\text{th bit of } C_P \text{ is } 1 \\ (\neg b_{c_j,k}), & \text{if the } k\text{th bit of } C_P \text{ is } 0 \end{cases}$$

Evaluation

Problem set:

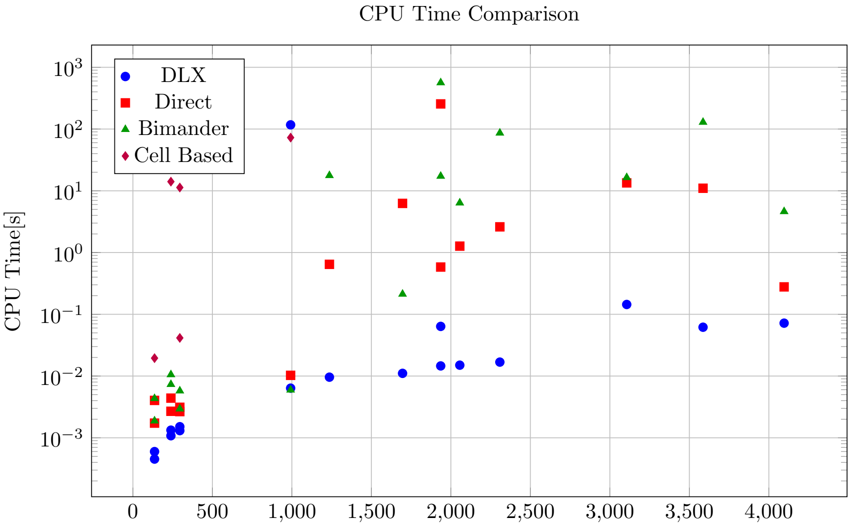
- Tetrominoes:
  - **Free**: 20 cells, 5 tetrominoes
    satisfiable:1
    unsatisfiable:1
  - **One-sided**: 28 cells, 7 tetrominoes
    satisfiable:2
    unsatisfiable:2
  - **Fixed**: 76 cells, 19 tetrominoes
    satisfiable:1
    unsatisfiable:1
- Pentominoes:
  - **Free**: 60 cells, 12 pentominoes
    satisfiable: 5
  - **One-sided**: 90 cells, 18 pentominoes
    satisfiable: 4

Evaluation

Metrics:

1. CPU runtime
2. Memory usage
3. Decisions
4. Propagations
5. Conflicts

Evaluation


CPU Time Comparison
DLX
Direct
Bimander
Cell Based
CPU Time[s]
10^3
10^2
10^1
10^0
10^{-1}
10^{-2}
10^{-3}
0
500
1,000
1,500
2,000
2,500
3,000
3,500
4,000

Evaluation

Approach	**Average Propagations per Decision**
DLX	47.76
Direct Encoding	12.00
Bimander Encoding	34.46
Cell-Based Encoding	17.64

Evaluation

Approach	**Average Conflicts per Decision**
DLX	5.11
Direct Encoding	0.22
Bimander Encoding	0.32
Cell-Based Encoding	0.67

Conclusion

- DLX outperforms all three SAT based encodings
- The more straightforward the encoding, the better it performs

DLX results

Instance	Conflicts	Decisions	Propagations	CPU time [s]	Memory [MB]
3x20 sat	1326	234	17352	0.009589	3.75
4x15 sat	1122	200	13682	0.011058	3.875
5x12 sat	5238	886	56449	0.014583	4
6x10 sat	4596	779	57459	0.015001	4
8x8-4 sat	6804	1147	67764	0.016817	4.125
10x10-10 sat	40224	6723	467223	0.07187	4.75
3x30 sat	64698	10802	727849	0.063681	4
5x18 sat	138948	23177	1685540	0.144209	4.375
9x10 sat	44856	7495	497206	0.061751	4.5
3x10-2 sat	460	100	2288	0.00108	3.375
3x10-2 unsat	1009	203	5239	0.00133	3.375
3x7-1 sat	5	7	147	0.000452	3.375
3x7-1 unsat	303	62	1313	0.000598	3.375
5x6-2 sat	180	44	1219	0.001307	3.375
5x6-2 unsat	1399	281	7045	0.001513	3.375
7x11-1 sat	1550	330	8887	0.006365	3.625
7x11-1 unsat	275101933	55020388	1448687742	116.949	3.375

Direct Encoding Results

Instance	Encoding		Search				
	CPU time [s]	Memory [MB]	Conflicts	Decisions	Propagations	CPU time [s]	Memory [MB]
3x20 s.	2.31265	7.171	11147	60737	652571	0.64443	25.29
4x15 s.	3.99141	10.992	38042	200155	3048119	6.2518	79.86
5x12 s.	4.92379	18.492	8710	69385	801152	0.582327	36.22
6x10 s.	5.4918	18.503	13944	112460	1529388	1.27028	48.37
8x8-4 s.	7.30714	18.492	24094	132569	1868586	2.60752	52.21
10x10-10 s.	37.0314	33.578	3046	33303	279499	0.277846	28.8
3x30 s.	7.58587	11.003	242731	943392	15723338	255.322	443.4
5x18 s.	19.4779	18.355	54038	263776	4427916	13.4272	176.95
9x10 s.	25.5386	33.53	46909	293838	4125333	10.9999	210.35
3x10-2 s.	0.046258	3.625	44	194	1024	0.00439	7.33
3x10-2 u.	0.044943	3.625	377	798	11123	0.002688	7.33
3x7-1 s.	0.010733	3.5	1	20	141	0.001729	7.07
3x7-1 u.	0.011638	3.5	18	62	790	0.004033	7.07
5x6-2 s.	0.065172	3.875	93	226	2564	0.002656	7.21
5x6-2 u.	0.069431	3.875	464	985	14368	0.003119	7.33
7x11-1 s.	1.81318	5.332	140	1048	7602	0.010265	9.01
7x11-1 u.	1.80924	5.328				>600	

Bimander Encoding Results

Instance	Encoding		Search				
	CPU time [s]	Memory [MB]	Conflicts	Decisions	Propagations	CPU time [s]	Memory [MB]
3x20 s.	0.046287	4.472	55099	212397	6409672	17.6958	85.44
4x15 s.	0.06737	4.468	3166	18137	367808	0.212215	14.46
5x12 s.	0.074659	5.464	49416	239880	8084070	17.2472	108.09
6x10 s.	0.079619	5.480	31333	141235	4400889	6.37287	77.14
8x8-4 s.	0.088903	5.464	131395	578411	18560493	86.2891	177.29
10x10-10 s.	0.228952	7.449	24407	116898	13886933	4.61235	70.45
3x30 s.	0.107787	5.476	602872	2358756	75332674	563.478	314.6
5x18 s.	0.181954	5.519	44799	205599	6597884	16.4028	144.48
9x10 s.	0.192946	7.375	145434	711812	21955936	129.049	544.99
3x10-2 s.	0.006126	3.625	749	1711	56556	0.010541	7.46
3x10-2 u.	0.006092	3.625	485	1692	30423	0.00733	7.34
3x7-1 s.	0.00358	3.5	56	118	2651	0.00188	7.07
3x7-1 u.	0.00357	3.5	117	171	6125	0.004332	7.07
5x6-2 s.	0.007108	3.625	132	383	7273	0.002939	7.2
5x6-2 u.	0.007489	3.375	408	778	27420	0.005759	7.21
7x11-1 s.	0.039523	4	226	691	18492	0.005949	7.77
7x11-1 u.	0.044513	3.875				>600	

Cell-Based Encoding Results

Instance	Encoding		Search				
	CPU time [s]	Memory [MB]	Conflicts	Decisions	Propagations	CPU time [s]	Memory [MB]
3x10-2 s.	0.023281	3.625	387611	545667	9958760	14.0702	19.2
3x10-2 u.	0.019283	3.625	1543883	2088415	34514864	57.6698	26.58
3x7-1 s.	0.008489	3.5	2186	3667	48861	0.019501	7.71
3x7-1 u.	0.006708	3.5	3829	5848	88787	0.041433	7.86
5x6-2 s.	0.024854	3.625	264268	405221	9399234	11.3036	21.44
5x6-2 u.	0.027779	3.375	1278494	1858781	35967381	72.7083	36.12
7x11-1 s.	0.471616	4				$>$600	
7x11-1 u.	0.480008	4				$>$600	