Reducing NEXP-complete problems to DQBF

Fa-Hsun Chen, Shen-Chang Huang, Yu-Cheng Lu, Tony Tan
National Taiwan University

Standard algorithm for NP-complete problems

- Transform it to a SAT instance and feed it to a SAT solver.

Standard algorithm for NP-complete problems

- Transform it to a SAT instance and feed it to a SAT solver.

Reasons:

- Modern SAT solvers are quite powerful.
- We have good intuition on how to reduce problems to SAT.

Standard algorithm for NP-complete problems

- Transform it to a SAT instance and feed it to a SAT solver.

Reasons:

- Modern SAT solvers are quite powerful.
- We have good intuition on how to reduce problems to SAT.

Cook-Levin reduction: One guessing step = one boolean variable.

Example: 3-colorability

(NP algorithm) On input graph G:

1. "Guess" the color of each vertex.
2. Verify that the coloring is proper.

Example: 3-colorability

(NP algorithm) On input graph G:

1. "Guess" the color of each vertex.
2. Verify that the coloring is proper.

(SAT encoding) On input graph G:

1. For each vertex v, we have boolean variables:

$$X_{v,R} \qquad X_{v,G} \qquad X_{v,B}$$

2. Construct a Boolean formula that encodes a proper coloring of G:

Example: 3-colorability

(NP algorithm) On input graph G:

1. "Guess" the color of each vertex.
2. Verify that the coloring is proper.

(SAT encoding) On input graph G:

1. For each vertex v, we have boolean variables:

$$X_{v,R} \qquad X_{v,G} \qquad X_{v,B}$$

2. Construct a Boolean formula that encodes a proper coloring of G:

$$\bigwedge_{v \in V} (X_{v,R} \vee X_{v,G} \vee X_{v,B}) \wedge \bigwedge_{v \in V} \bigwedge_{c \neq c'} X_{v,c} \to \neg X_{v,c'}$$

$$\wedge \bigwedge_{(u,v) \in E} \bigwedge_{c} \neg (X_{u,c} \wedge X_{v,c})$$

SAT is "the problem" in the class NP.

SAT is "the problem" in the class NP.

PSPACE: TQBF.

Reduction: Graph reachability encoded as QBF (Stockmeyer and Meyer, 1973).

SAT is "the problem" in the class NP.

PSPACE: TQBF.

Reduction: Graph reachability encoded as QBF (Stockmeyer and Meyer, 1973).

(Question) What is the problem for the class NEXP?

SAT is "the problem" in the class NP.

PSPACE: TQBF.

Reduction: Graph reachability encoded as QBF (Stockmeyer and Meyer, 1973).

(Question) What is the problem for the class NEXP?

(Papadimitriou and Yannakakis, 1986)

Well known NEXP-complete problems

Well known NEXP-complete problems

- Two-variable logic.

Well known NEXP-complete problems

- Two-variable logic.
- Bernays-Schönfinkel-Ramsey class.

Well known NEXP-complete problems

- Two-variable logic.
- Bernays-Schönfinkel-Ramsey class.
- Löwenheim/monadic class.

Well known NEXP-complete problems

- Two-variable logic.
- Bernays-Schönfinkel-Ramsey class.
- Löwenheim/monadic class.
- Succinctly represented NP-complete problems.

Well known NEXP-complete problems

- Two-variable logic.
- Bernays-Schönfinkel-Ramsey class.
- Löwenheim/monadic class.
- Succinctly represented NP-complete problems.
- Dependency Quantified Boolean Formulas (DQBF).

Well known NEXP-complete problems

- Two-variable logic.
- Bernays-Schönfinkel-Ramsey class.
- Löwenheim/monadic class.
- Succinctly represented NP-complete problems.
- Dependency Quantified Boolean Formulas (DQBF).
  (Exciting progress recently)

Well known NEXP-complete problems

- Two-variable logic.
- Bernays-Schönfinkel-Ramsey class.
- Löwenheim/monadic class.
- Succinctly represented NP-complete problems.
- Dependency Quantified Boolean Formulas (DQBF).
  (Exciting progress recently)

(Our contribution)

- A general method to reduce (in polynomial time) many interesting NEXP-complete problems to DQBF.
- It has a similar flavor to Cook-Levin reduction.

DQBF: A generalization of QBF

QBF:

$$\forall x_1 \forall x_2 \forall x_3 \; \exists y_1 \; \forall x_4 \forall x_5 \; \exists y_2 \quad \varphi$$

The value of y_1 is a function on x_1, x_2, x_3, i.e., depends on all x_1, x_2, x_3.

The value of y_2 is a function on $x_1, \ldots, x_5$.

DQBF: A generalization of QBF

QBF:

$$\forall x_1 \forall x_2 \forall x_3 \; \exists y_1 \; \forall x_4 \forall x_5 \; \exists y_2 \quad \varphi$$

The value of y_1 is a function on x_1, x_2, x_3, i.e., depends on all x_1, x_2, x_3.

The value of y_2 is a function on $x_1, \ldots, x_5$.

In DQBF we can specify the dependency of each existential variable.

$$\forall x_1 \forall x_2 \forall x_3 \; \exists y_1(x_1, x_3) \; \forall x_4 \forall x_5 \; \exists y_2(x_2, x_4, x_5) \quad \varphi$$

The value of y_1 is a function on x_1, x_3.

The value of y_2 is a function on x_2, x_4, x_5.

DQBF – Preliminaries

(DQBF normal form) Every DQBF can be transformed into:

$$\forall x_1 \cdots \forall x_n \; \exists y_1(\bar{z}_1) \cdots \exists y_m(\bar{z}_m) \quad \varphi$$

where each $\bar{z}_i \subseteq \{x_1, \ldots, x_n\}$.

DQBF – Preliminaries

(DQBF normal form) Every DQBF can be transformed into:

$$\forall x_1 \cdots \forall x_n \; \exists y_1(\bar{z}_1) \cdots \exists y_m(\bar{z}_m) \quad \varphi$$

where each $\bar{z}_i \subseteq \{x_1, \ldots, x_n\}$.

(Def.) It is *satisfiable* if for each i, there is a boolean function $f_i : \{0,1\}^{n_i} \to \{0,1\}$ such that φ is a tautology when y_i is replaced with $f_i(\bar{z}_i)$.

DQBF – Preliminaries

(DQBF normal form) Every DQBF can be transformed into:

$$\forall x_1 \cdots \forall x_n \; \exists y_1(\bar{z}_1) \cdots \exists y_m(\bar{z}_m) \quad \varphi$$

where each $\bar{z}_i \subseteq \{x_1, \ldots, x_n\}$.

(Def.) It is *satisfiable* if for each i, there is a boolean function $f_i : \{0,1\}^{n_i} \to \{0,1\}$ such that φ is a tautology when y_i is replaced with $f_i(\bar{z}_i)$.

Theorem (Patterson and Reif, 1979)
SAT(DQBF) is NEXP*-complete.*

Some work and solvers for DQBF

- V. Balabanov, H. K. Chiang, and J. R. Jiang, "Henkin quantifiers and boolean formulae: A certification perspective of DQBF," *Theor. Comput. Sci.*, vol. 523, pp. 86–100, 2014.
- A. Fröhlich, G. Kovásznai, and A. Biere, "A DPLL algorithm for solving DQBF," in *POS-12, Third Pragmatics of SAT workshop*, 2012.
- A. Ge-Ernst, C. Scholl, and R. Wimmer, "Localizing quantifiers for DQBF," in *FMCAD*, 2019.
- O. Kullmann and A. Shukla, "Autarkies for DQCNF," in *FMCAD*, 2019.
- R. Wimmer, C. Scholl, and B. Becker, "The (D)QBF preprocessor hqspre - underlying theory and its implementation," *J. Satisf. Boolean Model. Comput.*, vol. 11, no. 1, pp. 3–52, 2019.
- K. Wimmer, R. Wimmer, C. Scholl, and B. Becker, "Skolem functions for DQBF," in *ATVA*, 2016.
- R. Wimmer, S. Reimer, P. Marin, and B. Becker, "HQSpre – an effective preprocessor for QBF and DQBF," in *TACAS*, 2017.
- G. Kovásznai, "What is the state-of-the-art in DQBF solving," in *Join Conference on Mathematics and Computer Science*, 2016.
- C. Scholl and R. Wimmer, "Dependency quantified boolean formulas: An overview of solution methods and applications - extended abstract," in *SAT*, 2018.

Some work and solvers for DQBF (continued)

- A. Fröhlich, G. Kovásznai, A. Biere, and H. Veith, "iDQ: Instantiation-based DQBF solving," in *POS-14, Fifth Pragmatics of SAT workshop*, 2014.
- L. Tentrup and M. Rabe, "Clausal abstraction for DQBF," in *SAT*, 2019.
- K. Gitina, R. Wimmer, S. Reimer, M. Sauer, C. Scholl, and B. Becker, "Solving DQBF through quantifier elimination," in *DATE*, 2015.
- R. Wimmer, A. Karrenbauer, R. Becker, C. Scholl, and B. Becker, "From DQBF to QBF by dependency elimination," in *SAT*, 2017.
- J. Síc and J. Strejcek, "DQBDD: an efficient bdd-based DQBF solver," in *SAT*, 2021.
- .....

Some work and solvers for DQBF (continued)

- A. Fröhlich, G. Kovásznai, A. Biere, and H. Veith, "iDQ: Instantiation-based DQBF solving," in *POS-14, Fifth Pragmatics of SAT workshop*, 2014.
- L. Tentrup and M. Rabe, "Clausal abstraction for DQBF," in *SAT*, 2019.
- K. Gitina, R. Wimmer, S. Reimer, M. Sauer, C. Scholl, and B. Becker, "Solving DQBF through quantifier elimination," in *DATE*, 2015.
- R. Wimmer, A. Karrenbauer, R. Becker, C. Scholl, and B. Becker, "From DQBF to QBF by dependency elimination," in *SAT*, 2017.
- J. Síc and J. Strejcek, "DQBDD: an efficient bdd-based DQBF solver," in *SAT*, 2021.
- .....

We would like to use them for other NEXP-complete problems.

Example: Succinct 3-colorability

(Def.) A (boolean) circuit $C(\bar{x}_1, \bar{x}_2)$ represents a graph $G(C)$, where $\bar{x}_1, \bar{x}_2$ are vectors of n boolean variables:

- The set of vertices is $\{0,1\}^n$.
- (u, v) is an edge iff $C(u, v) = 1$.

Example: Succinct 3-colorability

(Def.) A (boolean) circuit $C(\bar{x}_1, \bar{x}_2)$ represents a graph $G(C)$, where $\bar{x}_1, \bar{x}_2$ are vectors of n boolean variables:

- The set of vertices is $\{0,1\}^n$.
- (u, v) is an edge iff $C(u, v) = 1$.

Succinct 3-colorability: On input circuit C, decide if the graph is 3-colorable.

Example: Succinct 3-colorability

(Def.) A (boolean) circuit $C(\bar{x}_1, \bar{x}_2)$ represents a graph $G(C)$, where $\bar{x}_1, \bar{x}_2$ are vectors of n boolean variables:

- The set of vertices is $\{0,1\}^n$.
- (u, v) is an edge iff $C(u, v) = 1$.

Succinct 3-colorability: On input circuit C, decide if the graph is 3-colorable.

Theorem (Papadimitriou and Yannakakis, 1986)
Succinct 3-colorability is NEXP*-complete.*

Reduction from succinct 3-colorability to SAT(DQBF)

(Main idea) On circuit $C(\bar{x}_1, \bar{x}_2)$:

- Represent the coloring with a function $f : \{0,1\}^n \to \{01, 10, 11\}$.
- The value of f is represented by a pair of existential variables (y_1, y_2).

Reduction from succinct 3-colorability to SAT(DQBF)

(Main idea) On circuit $C(\bar{x}_1, \bar{x}_2)$:

- Represent the coloring with a function $f : \{0,1\}^n \to \{01, 10, 11\}$.
- The value of f is represented by a pair of existential variables (y_1, y_2).

More formally:

$$\forall \bar{x}_1 \forall \bar{x}_2 \quad \exists y_1(\bar{x}_1) \exists y_2(\bar{x}_1) \quad \exists y_3(\bar{x}_2) \exists y_4(\bar{x}_2)$$

$$\bar{x}_1 = \bar{x}_2 \;\to\; (y_1, y_2) = (y_3, y_4) \tag{1}$$

$$\wedge\; (y_1, y_2) \neq (0,0) \;\wedge\; (y_3, y_4) \neq (0,0) \tag{2}$$

$$\wedge\; C(\bar{x}_1, \bar{x}_2) = 1 \;\to\; (y_1, y_2) \neq (y_3, y_4) \tag{3}$$

Reduction from succinct 3-colorability to SAT(DQBF)

(Main idea) On circuit $C(\bar{x}_1, \bar{x}_2)$:

- Represent the coloring with a function $f : \{0,1\}^n \to \{01, 10, 11\}$.
- The value of f is represented by a pair of existential variables (y_1, y_2).

More formally:

$$\forall \bar{x}_1 \forall \bar{x}_2 \quad \exists y_1(\bar{x}_1) \exists y_2(\bar{x}_1) \quad \exists y_3(\bar{x}_2) \exists y_4(\bar{x}_2)$$

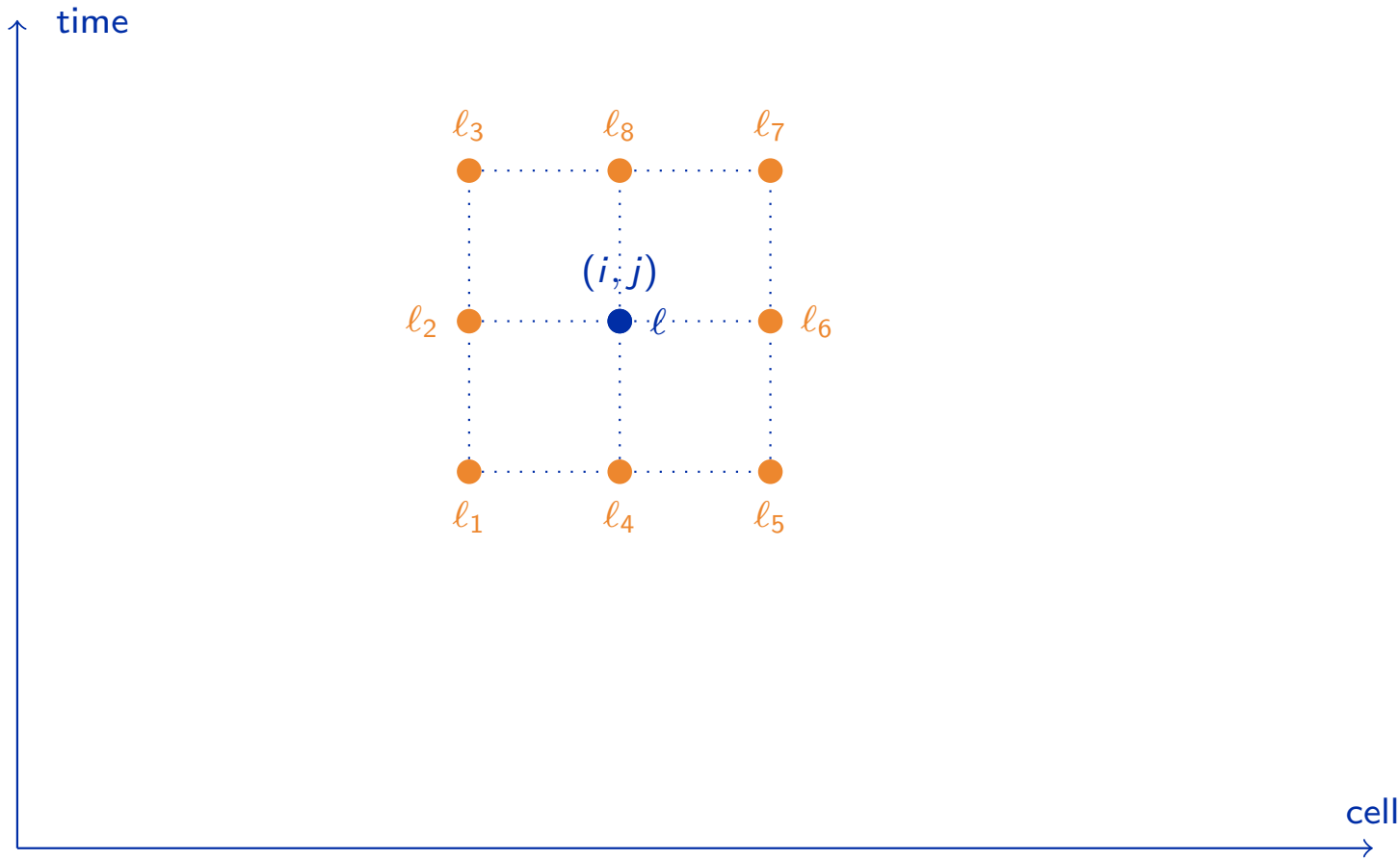
$$\bar{x}_1 = \bar{x}_2 \;\to\; (y_1, y_2) = (y_3, y_4) \tag{1}$$

$$\wedge\; (y_1, y_2) \neq (0,0) \;\wedge\; (y_3, y_4) \neq (0,0) \tag{2}$$

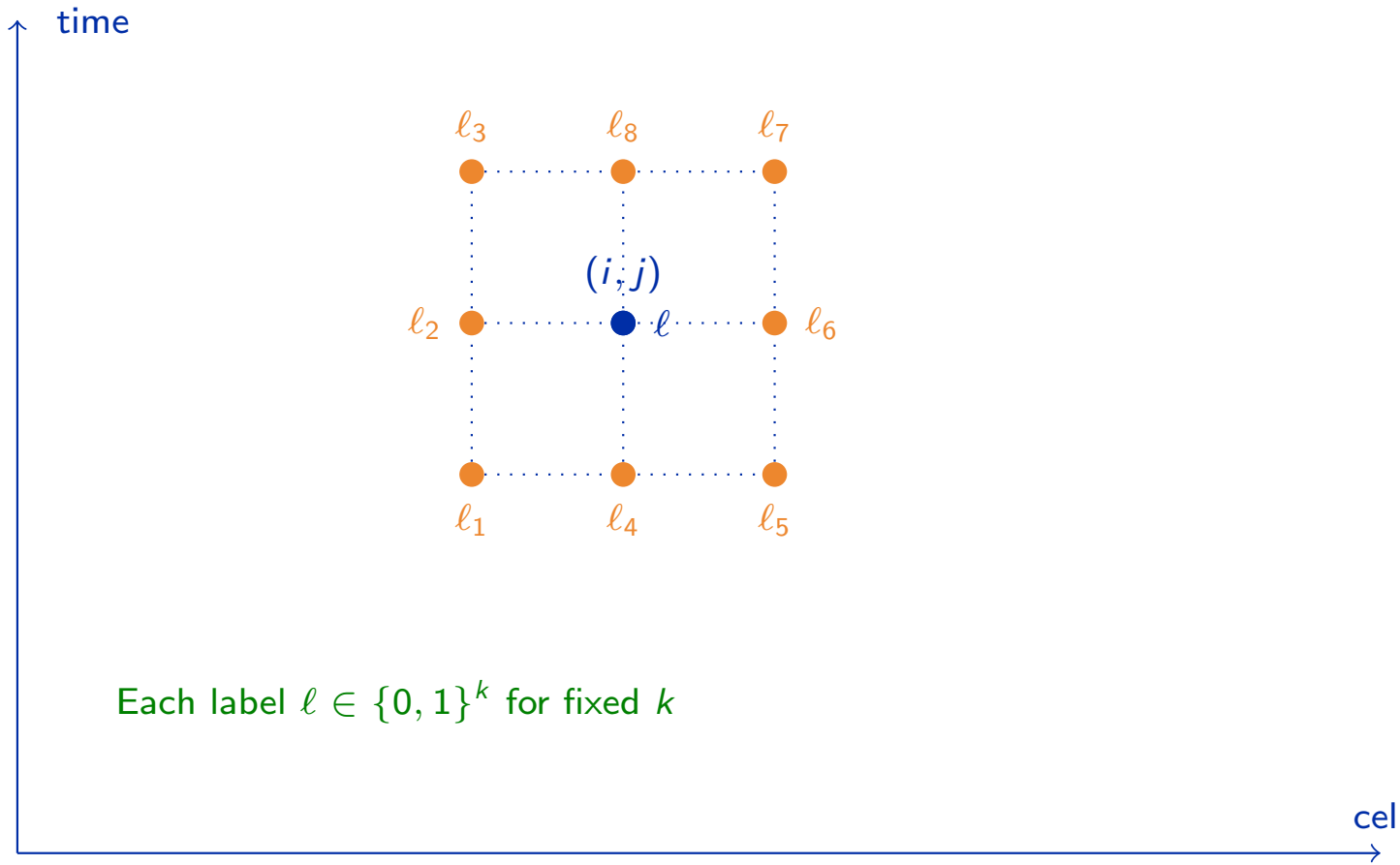
$$\wedge\; C(\bar{x}_1, \bar{x}_2) = 1 \;\to\; (y_1, y_2) \neq (y_3, y_4) \tag{3}$$

The reduction can be generalized to any NEXP-complete problems.

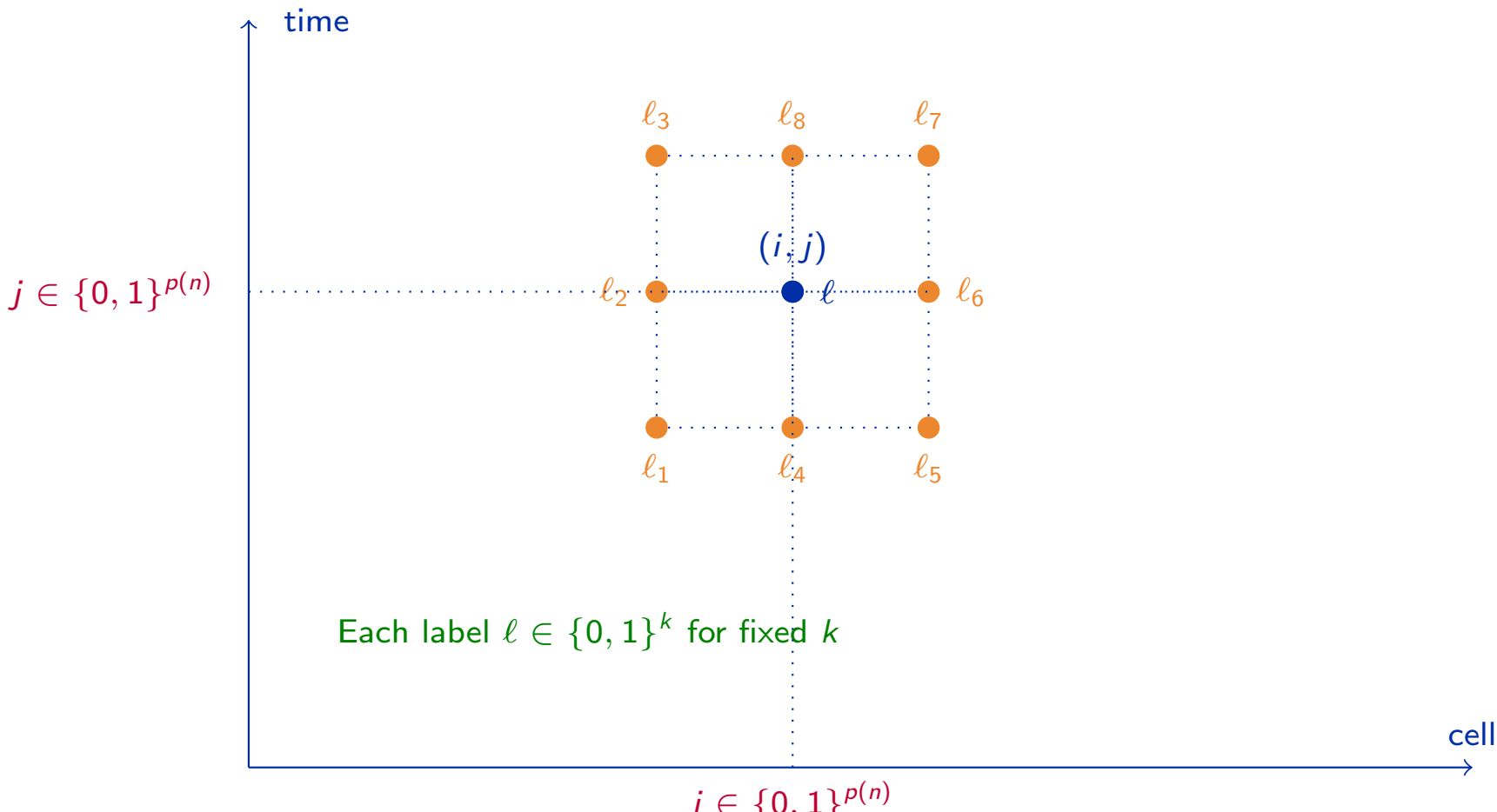
Main idea for general NEXP-complete problem: Succinct encoding of TM computation



Main idea for general NEXP-complete problem: Succinct encoding of TM computation

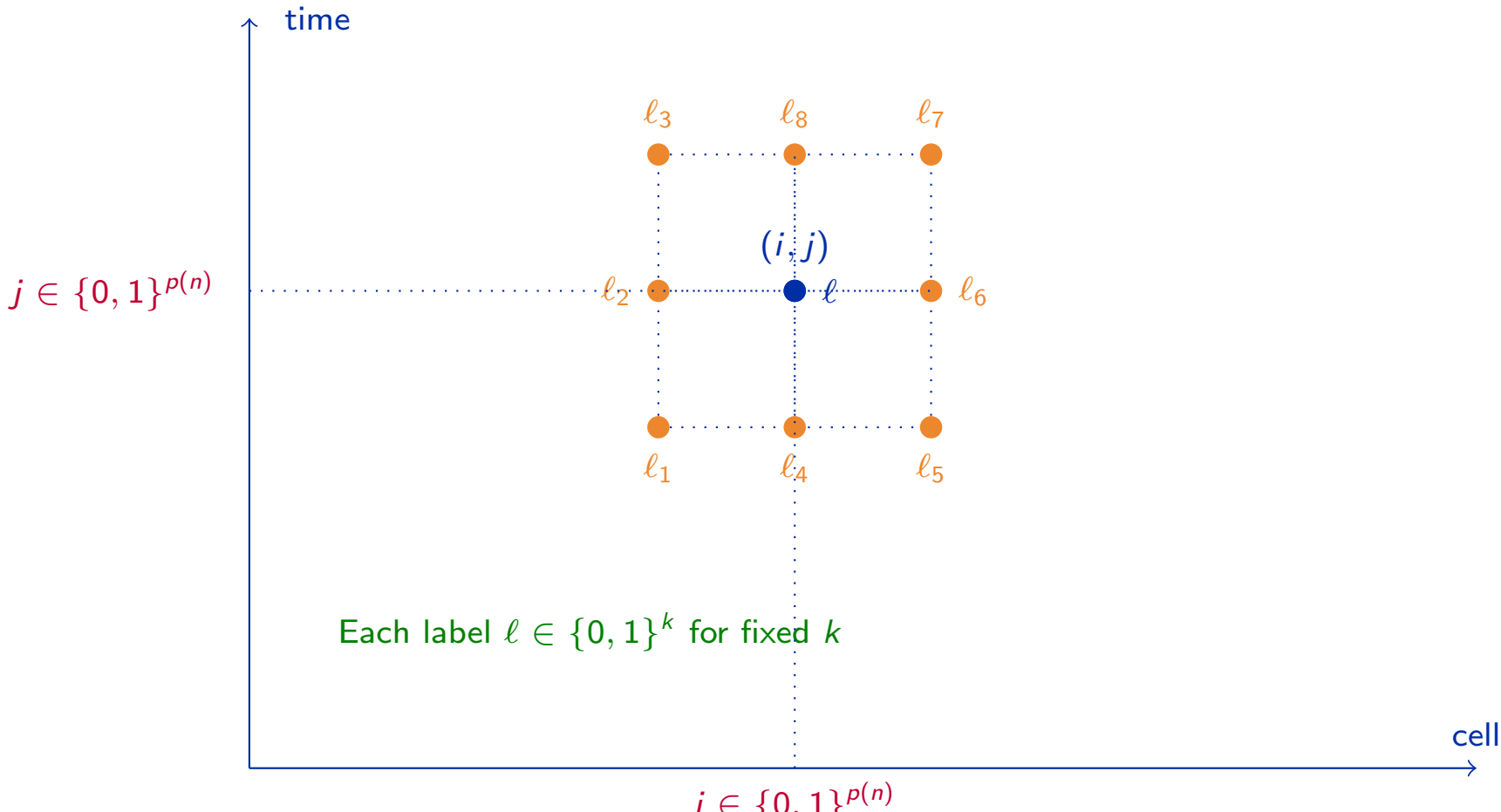


Main idea for general NEXP-complete problem: Succinct encoding of TM computation



Main idea for general NEXP-complete problem: Succinct encoding of TM computation

DQBF formula can state that the labels every two neigboring points "obey" the transitions of TM using $O(p(n))$ variables.



Formalization: Succinct projection

(Def.) For a circuit $C(\bar{x}_1, \bar{y}_1, \bar{x}_2, \bar{y}_2)$, where $|\bar{x}_1| = |\bar{x}_2| = n$ and $|\bar{y}_1| = |\bar{y}_2| = m$ a function $g : \{0,1\}^n \to \{0,1\}^m$ *agrees* with C, if:

$$C(\bar{u}, g(\bar{u}), \bar{v}, g(\bar{v})) = 1, \text{ for every } \bar{u}, \bar{v} \in \{0,1\}^m.$$

Intuitively, the function g represents the labeling of each point in the TM computation and C ensures that it obeys the TM transitions.

Formalization: Succinct projection

(Def.) For a circuit $C(\bar{x}_1, \bar{y}_1, \bar{x}_2, \bar{y}_2)$, where $|\bar{x}_1| = |\bar{x}_2| = n$ and $|\bar{y}_1| = |\bar{y}_2| = m$ a function $g : \{0,1\}^n \to \{0,1\}^m$ *agrees* with C, if:

$$C(\bar{u}, g(\bar{u}), \bar{v}, g(\bar{v})) = 1, \text{ for every } \bar{u}, \bar{v} \in \{0,1\}^m.$$

Intuitively, the function g represents the labeling of each point in the TM computation and C ensures that it obeys the TM transitions.

(Def.) A *succinct projection* for a language L is a polynomial time algorithm such that on input w, it outputs a circuit C:

$$w \in L \quad \text{iff} \quad \text{there is a function } g \text{ that agrees with } C.$$

The idea is that for NEXP languages the circuit C can be constructed in polynomial time.

Formalization: Succinct projection (continued)

Theorem

A language $L \in \mathrm{NEXP}$ iff it has a succinct projection.

Formalization: Succinct projection (continued)

Theorem

A language $L \in \mathrm{NEXP}$ iff it has a succinct projection.

(Reduction from $L \in$ NEXP to SAT(DQBF)) On input w:

1. Run the succinct projection on w to obtain the circuit C.
2. Output

$$\forall \bar{x}_1 \forall \bar{x}_2 \; \exists \bar{y}_1(\bar{z}_1) \exists \bar{y}_2(\bar{z}_2) \quad C(\bar{x}_1, \bar{y}_1, \bar{x}_2, \bar{y}_2) \; \wedge \; (\bar{x}_1 = \bar{x}_2 \rightarrow \bar{y}_1 = \bar{y}_2).$$

$\bar{y}_1$ and $\bar{y}_2$ represent the function that agrees with C (if exists).

In the paper (full version uploaded to ArXiv)

More reductions:

- (Succinct) independent set, Hamiltonian cycle. vertex cover, dominating set, set packing, subset sum, etc.
- Two-variable logic (FO^2).
- Bernays-Schönfinkel-Ramsey class (BSR).
- Löwenheim/monadic class.

In the paper (full version uploaded to ArXiv)

More reductions:

- (Succinct) independent set, Hamiltonian cycle. vertex cover, dominating set, set packing, subset sum, etc.
- Two-variable logic (FO^2).
- Bernays-Schönfinkel-Ramsey class (BSR).
- Löwenheim/monadic class.

With a little modification succinct projection does not apply only to DQBF, but can also be used to obtain reductions from any NEXP-complete problem to FO^2, BSR or Löwenheim/monadic class.

Our hope

- Richer benchmarks and applications of DQBF solvers.

Our hope

- Richer benchmarks and applications of DQBF solvers.
- Further development of the solvers.

Our hope

- Richer benchmarks and applications of DQBF solvers.
- Further development of the solvers.
- DQBF can be the *bona fide* problem in NEXP, just like SAT in NP and QBF in PSPACE.

Our hope

- Richer benchmarks and applications of DQBF solvers.
- Further development of the solvers.
- DQBF can be the *bona fide* problem in NEXP, just like SAT in NP and QBF in PSPACE.

Thank you very much!