SAT-Solving und Anwendungen
QSAT

Prof. Dr. Wolfgang Küchlin
Dipl. Inform. Christoph Zengler

Universität Tübingen

17. Juni 2009

Das QSAT Problem

SAT

Ist eine gegebene Formel in Aussagenlogik **erfüllbar** oder **nicht erfüllbar**?

QSAT

Ist eine gegebene **vollständig quantifizierte** Formel in Aussagenlogik **wahr** oder **falsch**?

Quantoren:

- $\exists x \; P$ Es existiert ein (oder mehrere) x, so dass Aussage P gilt
- $\forall x \; P$ Für alle x gilt Aussage P

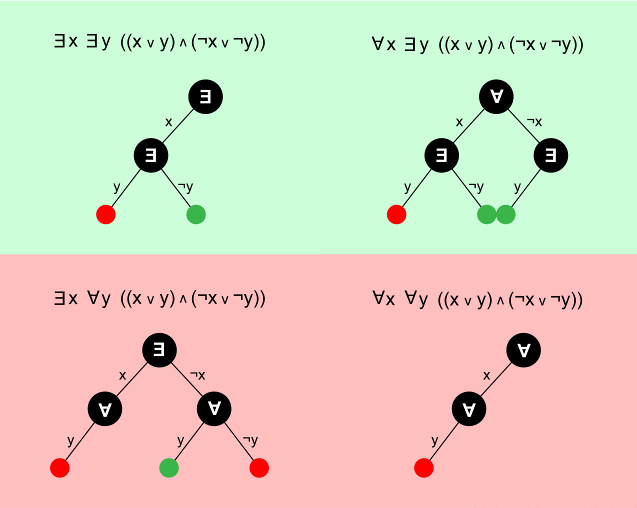
Beispiel (QSAT Probleme)

$$\forall x \exists y ((x \vee y) \wedge (\neg x \vee \neg y)) = \mathbf{T}$$

$$\forall x \forall y ((x \vee y) \wedge (\neg x \vee \neg y)) = \mathbf{F}$$

Visualisierung von QSAT

- 2 verschiedene Knotentypen: Existenzknoten und Allknoten
- Existenzknoten benötigen 1 erfüllenden Ast, Allknoten benötigen 2 erfüllende Äste



Komplexität von QSAT - 1

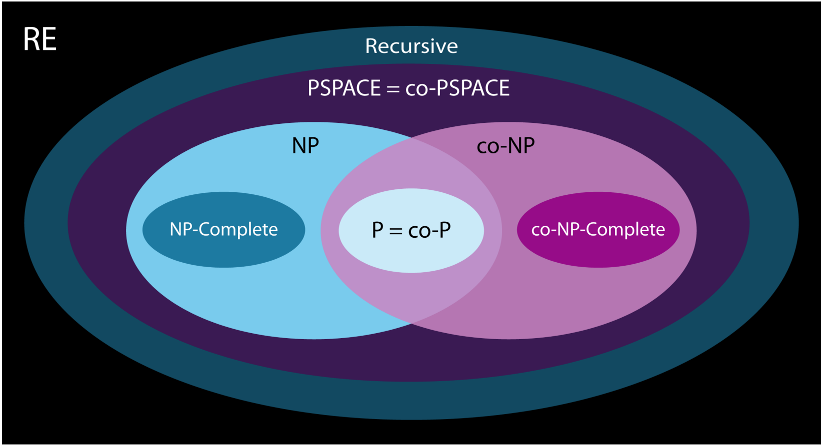
Im SAT Fall:

- NP-vollständig (Nichtdeterministische Turingmaschine, Polynomiale Zeit)
- Eine erfüllende Belegung kann geraten werden und in Polynomialzeit überprüft werden

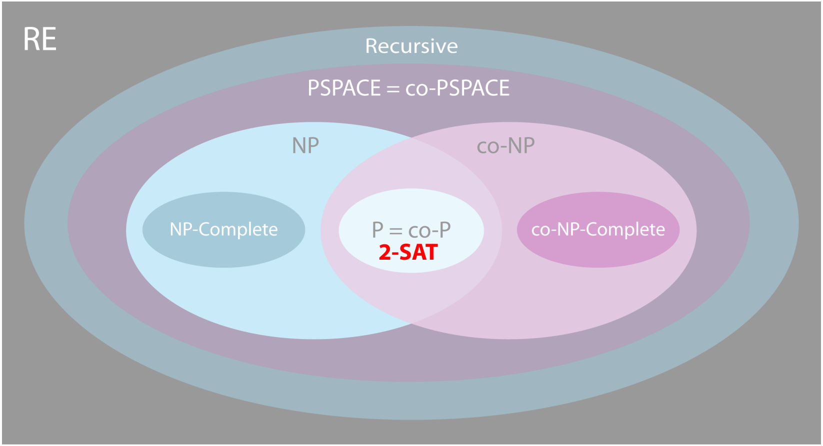
im QSAT Fall:

- PSPACE-vollständig (Terministische Turingmaschine, Polynomialer Platz)
- Es kann nicht mehr einfach eine Belegung angegeben werden, sondern man muss für jede mögliche Belegung der allquantifizierten Variablen eine Belegung der existensquantifizierten Variablen angeben

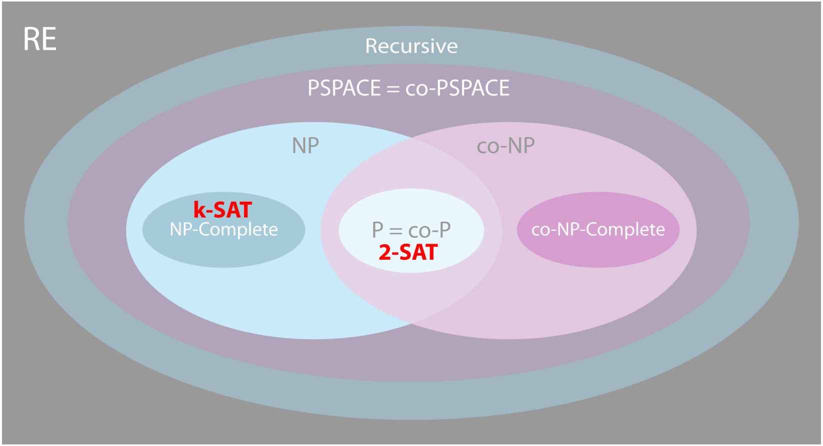
Komplexität von QSAT - 2



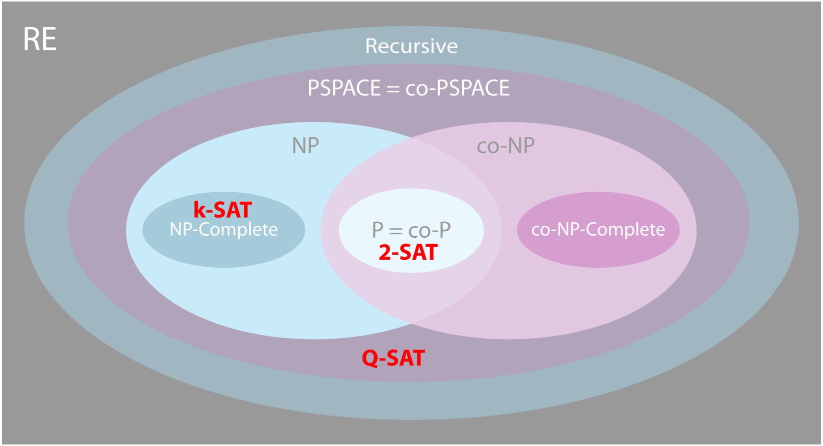
Komplexität von QSAT - 2



Komplexität von QSAT - 2



Komplexität von QSAT - 2



Formales

Pränexe Normal Form (PNF)

Eine quantifizierte boolesche Formel φ ist in PNF wenn Sie von der Form

$$Q_1 x_1, \ldots, Q_n x_n F$$

mit $Q_i \in \{\exists, \forall\}$ist und F quantorenfrei ist.

Jede Formel kann in PNF gebracht werden.

Freie / Gebundene Variablen

Eine Variable x kommt frei in einer Formel φ vor, wenn Sie nicht quantifiziert ist. Ist sie quantifiziert, so kommt sie gebunden vor.

Im QSAT Fall gibt es nur gebundene Variablen (Formel ist voll quantifiziert)

Vergleich der Algorithmen

- Backtracking auch im Fall, dass ein Ast erfüllend ist (bei Allquantoren)

SAT vs. QSAT Algorithmus

SAT Algorithmus

```
level := 0;
while true do
    unitPropgation();
    if a conflict is reached then
        level := analyseConflict();
        if level = 0 then
            return false
        end
        backtrack(level);
    else
        if formula is satisfied then
            return true
        end
        level := level + 1;
        choose an unassigned x ∈ var(P);
        α := α ∪ [x ↦ 0];
    end
end
```

QSAT Algorithmus

```
level := 0;
while true do
    unitPropgation();
    if a conflict is reached then
        level := analyseConflict();
        if level = 0 then
            return false
        end
        backtrack(level);
    else
        if formula is satisfied then
            level := analyseSAT();
            if level = 0 then
                return true
            end
            backtrack(level)
        else
            level := level + 1;
            choose an unassigned x ∈ var(P)
            (wrt. the q-level);
            α := α ∪ [x ↦ 0];
        end
    end
end
```

Auswahlheuristik

- Prinzipiell die selben Heuristiken wie im SAT Fall
- Müssen Quantifikations Level beachten
- Quantifikations Level wird mit jedem Quantorenwechsel erhöht

Beispiel (Quantifikations Level)

$$\underbrace{\exists x \; \exists y}_{level 1} \; \underbrace{\forall z \; \forall w}_{level 2} \; \underbrace{\exists u}_{level 3} (x \vee y \vee z \vee w \vee u)$$

- Heuristik muss von außen nach innen voranschreiten (d.h von level 1 aufwärts)
- Solange noch Variablen auf einem Level n nicht belegt sind, darf keine Variable auf einem Level $> n$ gewählt werden (gilt nicht für UP)
- Worst case: $\exists x_1 \; \forall x_2 \; \exists x_3 \forall x_4 \ldots F$ (Keine Wahlmöglichkeiten)

Neue Regel für empty clauses

Im SAT Fall: Eine Klausel ist unerfüllbar (empty clause) wenn sie noch nicht erfüllt ist und alle Variablen belegt sind.

Neue Regel im QSAT Fall:

- $E(C)$ existenzquantifizierte Variablen einer Klausel C
- $U(C)$ allquantifizierte Variablen einer Klausel C
- $ql(x)$ Quantifikationslevel einer Variable x

Empty clause

Eine Klausel C ist unerfüllbar (empty clause), wenn

1. für alle $e \in E(C)$ gilt $\nu(e) = \bot$
2. für alle $u \in U(C)$ gilt $\nu(u) \neq \top$

Beispiel (Empty clause)

a, b, c sind existenzquantifiziert, x, y sind allquantifiziert

$[a \mapsto \top, b \mapsto \bot, c \mapsto \top, x \mapsto \bot,]$

- $(\neg a \vee b \vee \neg c \vee x \vee y)$ ist unerfüllbar

Neue Regel für unit clauses

Regel im SAT Fall: Eine Klausel ist unit, wenn sie noch nicht erfüllt ist und genau eine Variable nicht belegt ist.

Neue Regel im QSAT Fall:

Unit clause

Eine Klausel C ist unit, wenn

1. ein $e \in E(C)$ existiert, so dass gilt $\nu(e) = nil$.
2. Für jedes $e' \in E(C), e' \neq e$ gilt, dass $\nu(e') = \bot$
3. Für alle $u \in U(C)$ mit $\nu(u) \neq \top$ gilt, dass $\nu(u) = nil \Rightarrow ql(u) > ql(e)$

Beispiel (Unit Clause)

a, b, c sind existenzquantifiziert, x, y sind allquantifiziert

$[a \mapsto \bot, c \mapsto \top, x \mapsto \bot]$, für die nächste durch UP implizierte Variable b gilt $ql(b) = 5$

- $(a_{(2)} \vee b_{(5)} \vee \neg c_{(3)} \vee x_{(1)} \vee y_{(6)})$ ist unit
- $(a_{(2)} \vee b_{(5)} \vee \neg c_{(3)} \vee x_{(4)} \vee y_{(1)})$ ist nicht unit

- Nur existenzquantifizierte Variablen können durch UP impliziert werden

Backtracking für eine erfüllende Belegung

- Für allquantifizierte Variablen müssen beide Belegungen getestet werden
- Jede allquantifizierte Variable muss geflipped werden
- Jede allquantifizierte Variable bekommt ein flag “flipped”
- Wird x belegt, so wird das flag auf `false` gesetzt
- Wird der Wert von x geflipped, wird das flag auf `true` gesetzt

Backtracking im erfüllenden Fall

Suche die letzte allquantifizierte Variable x, deren flag `false` ist, mache ein Backtracking zum level von x und flippe den Wert von x.

Lernen in QSAT

Funktioniert im Prinzip wie bei SAT, aber 2 Besonderheiten

- Es kann zu Long Distance Resolutions kommen
- Modifiziertes Stop Kriterium für 1UIP

Long Distance Resolution

- Resolution bei SAT: Nur 1 Variable, dass sich im Vorzeichen unterscheidet (Distance 1)
- Bei QSAT: Mehrere Variablen können sich im Vorzeichen unterscheiden

Grund: Allquantifizierte Variablen, die noch nicht belegt sind.

Beispiel (Long Distance Resolution)

$$(a_{(1)} \vee b_{(3)} \vee x_{(4)} \vee y_{(4)} \vee c_{(5)}) \wedge (a_{(1)} \vee \neg b_{(3)} \vee \neg x_{(4)} \vee \neg y_{(4)} \vee d_{(5)})$$

Resolution zwischen beiden Klauseln ist eine Tautologie:

$$(a_{(1)} \vee x_{(4)} \vee \neg x_{(4)} \vee y_{(4)} \vee \neg y_{(4)} \vee c_{(5)} \vee d_{(5)})$$

Aber Klauseln haben zwei Zwecke:

- Erkennen von empty clauses
- Erkennen von unit clauses

$\rightarrow$ Beides funktioniert mit Tautologieklauseln

Stop Kriterium für 1UIP

Ziel von 1UIP:

- neu gelernte Klausel soll nach Backtracking unit sein

Wegen geändertem Kriterium für unit clauses muss auch das Stop Kriterium für 1UIP angepasst werden

Stop Kriterium für 1UIP

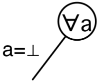
1. Nur eine existenzquantifizierte Variable e ist auf höchstem Level
2. e ist auf einem decision level wo die Entscheidungsvariable existenzquantifiziert ist
3. Alle allquantifizierten Variablen u mit $ql(u) < ql(e)$ werden zu $\perp$ evaluiert auf einem decision level $<$ dem von e

Wie bei SAT: Backtracking zu größtem Level, dass $<$ dem decision level von e ist.

Ein Beispiel für QSAT

$\forall a \exists x \exists y \forall b \exists z$

{x,y,z,¬b}
{x,¬z,a}
{¬x,y,a}
{x,¬y,z,b}

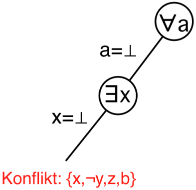


Level	Var	Quant	Wert	Grund
1	a	∀(1) F	⊥	decision

Ein Beispiel für QSAT

$\forall a \exists x \exists y \forall b \exists z$

$\{x, y, z, \neg b\}$
$\{x, \neg z, a\}$
$\{\neg x, y, a\}$
$\{x, \neg y, z, b\}$

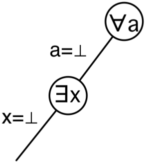


Level	Var	Quant	Wert	Grund
1	a	$\forall(1)$ F	$\perp$	decision
2	x	$\exists(2)$	$\perp$	decision
	z	$\exists(4)$	$\perp$	$\{x, \neg z, a\}$
	y	$\exists(2)$	$\top$	$\{x, y, z, \neg b\}$

Ein Beispiel für QSAT

$\forall a \exists x \exists y \forall b \exists z$

$\{x,y,z,\neg b\}$
$\{x,\neg z,a\}$
$\{\neg x,y,a\}$
$\{x,\neg y,z,b\}$



Konflikt: $\{x,\neg y,z,b\}$

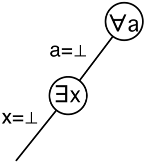
Level	Var	Quant	Wert	Grund
1	a	$\forall(1)$ F	$\perp$	decision
2	x	$\exists(2)$	$\perp$	decision
	z	$\exists(4)$	$\perp$	$\{x,\neg z,a\}$
	y	$\exists(2)$	$\top$	$\{x,y,z,\neg b\}$

$\{x_2,\neg y_2,z_2,b_{na}\}$ $\{x_2,y_2,z_2,\neg b_{na}\}$

Ein Beispiel für QSAT

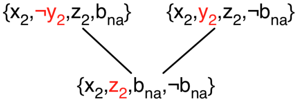
$\forall a \exists x \exists y \forall b \exists z$

$\{x, y, z, \neg b\}$
$\{x, \neg z, a\}$
$\{\neg x, y, a\}$
$\{x, \neg y, z, b\}$



Konflikt: $\{x, \neg y, z, b\}$

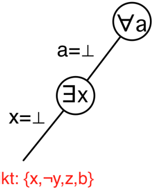
Level	Var	Quant	Wert	Grund
1	a	$\forall(1)$ F	$\perp$	decision
2	x	$\exists(2)$	$\perp$	decision
	z	$\exists(4)$	$\perp$	$\{x, \neg z, a\}$
	y	$\exists(2)$	$\top$	$\{x, y, z, \neg b\}$



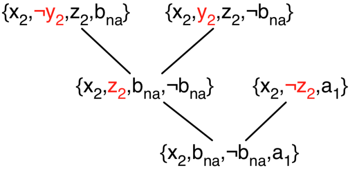
Ein Beispiel für QSAT

$\forall a \exists x \exists y \forall b \exists z$

{x,y,z,¬b}
{x,¬z,a}
{¬x,y,a}
{x,¬y,z,b}



Level	Var	Quant	Wert	Grund
1	a	∀(1) F	⊥	decision
2	x	∃(2)	⊥	decision
	z	∃(4)	⊥	{x,¬z,a}
	y	∃(2)	⊤	{x,y,z,¬b}

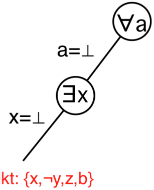


Ein Beispiel für QSAT

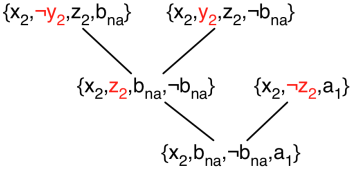
$\forall a \exists x \exists y \forall b \exists z$

{x,y,z,¬b}
{x,¬z,a}
{¬x,y,a}
{x,¬y,z,b}

{x,b,¬b,a}



Level	Var	Quant	Wert	Grund
1	a	∀(1) F	⊥	decision
2	x	∃(2)	⊥	decision
	z	∃(4)	⊥	{x,¬z,a}
	y	∃(2)	⊤	{x,y,z,¬b}

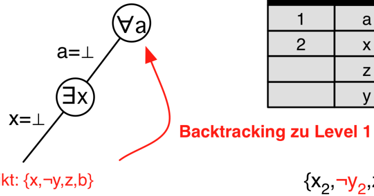


Ein Beispiel für QSAT

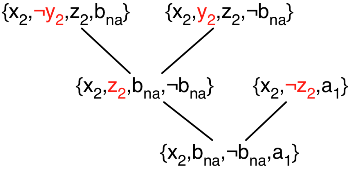
$\forall a \exists x \exists y \forall b \exists z$

$\{x,y,z,\neg b\}$
$\{x,\neg z,a\}$
$\{\neg x,y,a\}$
$\{x,\neg y,z,b\}$

$\{x,b,\neg b,a\}$



Level	Var	Quant	Wert	Grund
1	a	$\forall(1)$ F	$\perp$	decision
2	x	$\exists(2)$	$\perp$	decision
	z	$\exists(4)$	$\perp$	$\{x,\neg z,a\}$
	y	$\exists(2)$	$\top$	$\{x,y,z,\neg b\}$

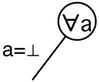


Ein Beispiel für QSAT

$\forall a \exists x \exists y \forall b \exists z$

$\{x,y,z,\neg b\}$
$\{x,\neg z,a\}$
$\{\neg x,y,a\}$
$\{x,\neg y,z,b\}$

$\{x,b,\neg b,a\}$



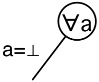
Level	Var	Quant	Wert	Grund
1	a	∀(1) F	⊥	decision

Ein Beispiel für QSAT

$\forall a \exists x \exists y \forall b \exists z$

{x,y,z,¬b}
{x,¬z,a}
{¬x,y,a}
{x,¬y,z,b}

{x,b,¬b,a}



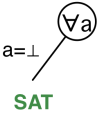
Level	Var	Quant	Wert	Grund
1	a	∀(1) F	⊥	decision
	x	∃(2)	⊤	{x,b,¬b,a}
	y	∃(2)	⊤	{¬x,y,a}

Ein Beispiel für QSAT

$\forall a \exists x \exists y \forall b \exists z$

{x,y,z,¬b}
{x,¬z,a}
{¬x,y,a}
{x,¬y,z,b}

{x,b,¬b,a}



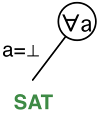
Level	Var	Quant	Wert	Grund
1	a	∀(1) F	⊥	decision
	x	∃(2)	⊤	{x,b,¬b,a}
	y	∃(2)	⊤	{¬x,y,a}

Ein Beispiel für QSAT

$\forall a \exists x \exists y \forall b \exists z$

{x,y,z,¬b}
{x,¬z,a}
{¬x,y,a}
{x,¬y,z,b}

{x,b,¬b,a}



Level	Var	Quant	Wert	Grund
1	a	∀(1) F	⊥	decision
	x	∃(2)	⊤	{x,b,¬b,a}
	y	∃(2)	⊤	{¬x,y,a}

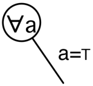
Backtracking zur letzten Variable mit Flag F

Ein Beispiel für QSAT

$\forall a \exists x \exists y \forall b \exists z$

$\{x, y, z, \neg b\}$
$\{x, \neg z, a\}$
$\{\neg x, y, a\}$
$\{x, \neg y, z, b\}$

$\{x, b, \neg b, a\}$



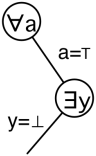
Level	Var	Quant	Wert	Grund
1	a	$\forall(1)\ \top$	$\top$	decision

Ein Beispiel für QSAT

$\forall a \exists x \exists y \forall b \exists z$

$\{x,y,z,\neg b\}$
$\{x,\neg z,a\}$
$\{\neg x,y,a\}$
$\{x,\neg y,z,b\}$

$\{x,b,\neg b,a\}$



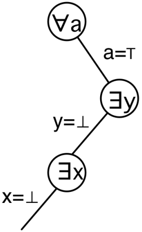
Level	Var	Quant	Wert	Grund
1	a	$\forall(1)$ ⊤	⊤	decision
2	y	$\exists(2)$	⊥	decision

Ein Beispiel für QSAT

$\forall a \exists x \exists y \forall b \exists z$

{x,y,z,¬b}
{x,¬z,a}
{¬x,y,a}
{x,¬y,z,b}

{x,b,¬b,a}



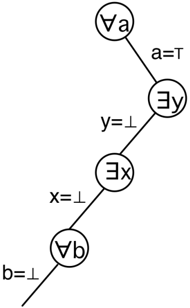
Level	Var	Quant	Wert	Grund
1	a	$\forall(1)$ ⊤	⊤	decision
2	y	$\exists(2)$	⊥	decision
3	x	$\exists(2)$	⊥	decision

Ein Beispiel für QSAT

$\forall a \exists x \exists y \forall b \exists z$

$\{x, y, z, \neg b\}$
$\{x, \neg z, a\}$
$\{\neg x, y, a\}$
$\{x, \neg y, z, b\}$

$\{x, b, \neg b, a\}$



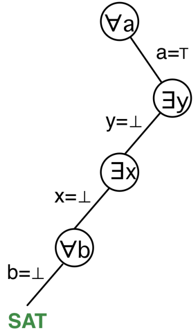
Level	Var	Quant	Wert	Grund
1	a	$\forall(1)$ T	$\top$	decision
2	y	$\exists(2)$	$\bot$	decision
3	x	$\exists(2)$	$\bot$	decision
4	b	$\forall(3)$ F	$\bot$	decision

Ein Beispiel für QSAT

$\forall a \exists x \exists y \forall b \exists z$

$\{x, y, z, \neg b\}$
$\{x, \neg z, a\}$
$\{\neg x, y, a\}$
$\{x, \neg y, z, b\}$

$\{x, b, \neg b, a\}$



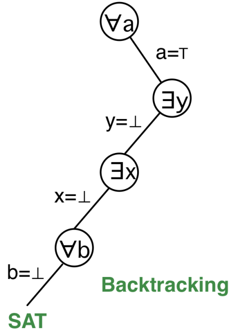
Level	Var	Quant	Wert	Grund
1	a	$\forall(1)$ T	$\top$	decision
2	y	$\exists(2)$	$\bot$	decision
3	x	$\exists(2)$	$\bot$	decision
4	b	$\forall(3)$ F	$\bot$	decision

Ein Beispiel für QSAT

$\forall a \exists x \exists y \forall b \exists z$

{x,y,z,¬b}
{x,¬z,a}
{¬x,y,a}
{x,¬y,z,b}

{x,b,¬b,a}



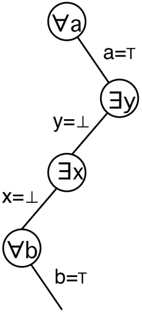
Level	Var	Quant	Wert	Grund
1	a	$\forall$(1) T	⊤	decision
2	y	$\exists$(2)	⊥	decision
3	x	$\exists$(2)	⊥	decision
4	b	$\forall$(3) F	⊥	decision

Ein Beispiel für QSAT

$\forall a \exists x \exists y \forall b \exists z$

$\{x,y,z,\neg b\}$
$\{x,\neg z,a\}$
$\{\neg x,y,a\}$
$\{x,\neg y,z,b\}$

$\{x,b,\neg b,a\}$



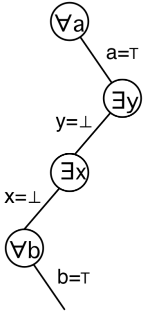
Level	Var	Quant	Wert	Grund
1	a	$\forall(1)$ T	$\top$	decision
2	y	$\exists(2)$	$\bot$	decision
3	x	$\exists(2)$	$\bot$	decision
4	b	$\forall(3)$ T	$\top$	decision

Ein Beispiel für QSAT

$\forall a \exists x \exists y \forall b \exists z$

$\{x,y,z,\neg b\}$
$\{x,\neg z,a\}$
$\{\neg x,y,a\}$
$\{x,\neg y,z,b\}$

$\{x,b,\neg b,a\}$



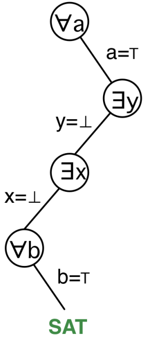
Level	Var	Quant	Wert	Grund
1	a	$\forall(1)$ ⊤	⊤	decision
2	y	$\exists(2)$	⊥	decision
3	x	$\exists(2)$	⊥	decision
4	b	$\forall(3)$ ⊤	⊤	decision
	z	$\exists(4)$	⊤	$\{x,y,z,\neg b\}$

Ein Beispiel für QSAT

$\forall a \exists x \exists y \forall b \exists z$

$\{x, y, z, \neg b\}$
$\{x, \neg z, a\}$
$\{\neg x, y, a\}$
$\{x, \neg y, z, b\}$

$\{x, b, \neg b, a\}$



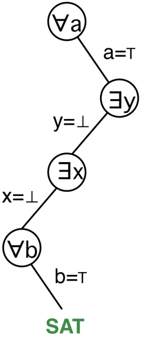
Level	Var	Quant	Wert	Grund
1	a	$\forall(1)$ $\top$	$\top$	decision
2	y	$\exists(2)$	$\bot$	decision
3	x	$\exists(2)$	$\bot$	decision
4	b	$\forall(3)$ $\top$	$\top$	decision
	z	$\exists(4)$	$\top$	$\{x, y, z, \neg b\}$

Ein Beispiel für QSAT

$\forall a \exists x \exists y \forall b \exists z$

{x,y,z,¬b}
{x,¬z,a}
{¬x,y,a}
{x,¬y,z,b}

{x,b,¬b,a}



Level	Var	Quant	Wert	Grund
1	a	∀(1) ⊤	⊤	decision
2	y	∃(2)	⊥	decision
3	x	∃(2)	⊥	decision
4	b	∀(3) ⊤	⊤	decision
	z	∃(4)	⊤	{x,y,z,¬b}

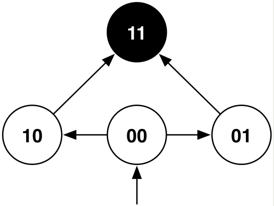
Keine weiteren Flags mit F

➔ Result: TRUE

Codierung mit QSAT - 1

Beispiel (Endlicher Automat)

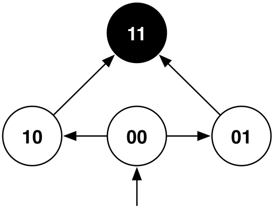
- 4 Zustände: $00, 01, 10, 11$
- Startzustand: 00
- Zustandsübergänge: $00 \rightarrow 01, 00 \rightarrow 10, 10 \rightarrow 11, 01 \rightarrow 11$
- Fehlerzustand: 11



Codierung:

- $s[0]$ und $s[1]$ codieren die beiden Bits des Zustands
- tiefergestellte Zahlen codieren den aktuellen timestep $s[0]_0$

Codierung mit QSAT - 2

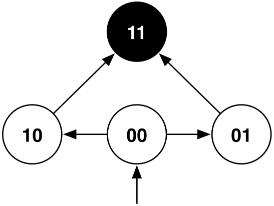


Fragestellung: Erreicht man einen Fehlerzustand in einem Schritt?

- Initialer Zustand: $I(0) = \neg s[0]_0 \wedge \neg s[1]_0$
- Übergangsfunktion:
  $T(0,1) = (\neg s[0]_0 \wedge \neg s[1]_0 \wedge \neg s[0]_1 \wedge s[1]_1) \vee (\neg s[0]_0 \wedge \neg s[1]_0 \wedge s[0]_1 \wedge \neg s[1]_1) \vee (\neg s[0]_0 \wedge s[1]_0 \wedge s[0]_1 \wedge s[1]_1) \vee (s[0]_0 \wedge \neg s[1]_0 \wedge s[0]_1 \wedge s[1]_1)$
- Fehlerzustand nach einem Schritt: $B(1) = s[0]_1 \wedge s[1]_1$

Checke $I(0) \wedge T(0,1) \wedge B(1)$, wenn erfüllbar, dann ist die Belegung ein Pfad von 00 zu 11, ansonsten gibt es keinen Pfad der Länge 1 zum Fehler

Codierung mit QSAT - 3

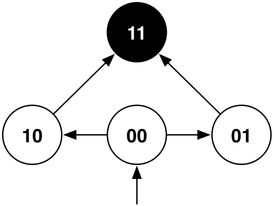


Fragestellung: Erreicht man einen Fehlerzustand in zwei Schritten?

- Initialer Zustand: $I(0) = \neg s[0]_0 \wedge \neg s[1]_0$
- Übergangsfunktion: $T(0,1) = (\neg s[0]_0 \wedge \neg s[1]_0 \wedge \neg s[0]_1 \wedge s[1]_1) \vee (\neg s[0]_0 \wedge \neg s[1]_0 \wedge s[0]_1 \wedge \neg s[1]_1) \vee (\neg s[0]_0 \wedge s[1]_0 \wedge s[0]_1 \wedge s[1]_1) \vee (s[0]_0 \wedge \neg s[1]_0 \wedge s[0]_1 \wedge s[1]_1)$
  $T(1,2) = (\neg s[0]_1 \wedge \neg s[1]_1 \wedge \neg s[0]_2 \wedge s[1]_2) \vee (\neg s[0]_1 \wedge \neg s[1]_1 \wedge s[0]_2 \wedge \neg s[1]_2) \vee (\neg s[0]_1 \wedge s[1]_1 \wedge s[0]_2 \wedge s[1]_2) \vee (s[0]_1 \wedge \neg s[1]_1 \wedge s[0]_2 \wedge s[1]_2)$
- Fehlerzustand nach zwei Schritten: $B(2) = s[0]_2 \wedge s[1]_2$

- Erfüllbar (00 – 01 – 11 oder 00 – 10 – 11 als Pfad zum Fehler)
- **Aber:** zwei Kopien der Übergangsfunktion

Codierung mit QSAT - 4



Codierung des Problems mit QSAT:

- $T = (\neg u[0] \wedge \neg u[1] \wedge \neg v[0] \wedge v[1]) \vee (\neg u[0] \wedge \neg u[1] \wedge v[0] \wedge \neg v[1]) \vee (\neg u[0] \wedge u[1] \wedge v[0] \wedge v[1]) \vee (u[0] \wedge \neg u[1] \wedge v[0] \wedge v[1])$

Formel für k Schritte

$\exists s[0]_0 \ldots \exists s[0]_k \exists s[1]_0 \ldots \exists s[1]_k \forall u[0] \forall u[1] \forall v[0] \forall v[1]$

$$I(0) \wedge B(k) \wedge \bigvee_{i=0}^{k-1} \left(u[0] \leftrightarrow s[0]_i \wedge u[1] \leftrightarrow s[1]_i \wedge v[0] \leftrightarrow s[0]_{i+1} \wedge v[1] \leftrightarrow s[1]_{i+1} \right) \rightarrow T$$

Nur noch eine Kopie der Übergangsfunktion T