The lazy programmer’s approach

Allows us to implement problem solving programs rapidly

We want to model a problem by compiling it into a suitable representation s.t. the result of the compiled problem can be interpreted as a solution to the original problem.
The lazy programmer’s approach

Allows us to implement problem solving programs rapidly

We want to model a problem by compiling it into a suitable representation s.t. the result of the compiled problem can be interpreted as a solution to the original problem.
The lazy programmer’s approach

Allows us to implement problem solving programs rapidly

We want to model a problem by compiling it into a suitable representation s.t. the result of the compiled problem can be interpreted as a solution to the original problem.
The topics for today

- Normal form translation again
  - What are the difficulties with prenexing?
  - What to do with the propositional matrix?

- How to implement a QSAT solver?
  - Only few remarks on QDPLL but
  - More on alternative and less explored possibilities
    - Q-resolution calculus
    - Gentzen (sequent) systems
Normal forms again
Prenex normal form (PNF), prefix, matrix, PCNF, closed

- Let $Q_i \in \{\forall, \exists\}$ and $p_i \in \mathcal{P}$. A QBF

$$\Phi = Q_1 p_1 \ldots Q_n p_n \psi$$

is in prenex (normal) form (PNF) if $\psi$ is purely propositional
- $Q_1 p_1 Q_2 p_2 \ldots Q_n p_n$ is the prefix of $\Phi$; $\psi$ is the matrix of $\Phi$.
- $\Phi$ is in PCNF if $\psi$ is in CNF
- $\Phi$ is closed if the variables in $\psi$ are in $\{p_1, \ldots, p_n\}$

Convention: Each quantifier binds another variable and bound variables do not occur free.
Generating PCNFs

Why are formulas in PCNF necessary?

- Most QBF solvers require the input being in PCNF
- Translation procedure required
  - Generate a prenex form
  - From the matrix, generate a CNF using Tseitin
- All steps here are equivalence-preserving (why?) (in contrast to, e.g., propositional logic)
Generating a prenex form (cf predicate logic)

Apply the following rules until a PNF is obtained

\[ R_1 \quad Qx \, \Phi \circ Qy \, \Psi \implies QxQy \,(\Phi \circ \Psi) \quad x \text{ not free in } \Psi, \, y \text{ not free in } \Phi \]

\[ R_2 \quad (Qx \, \Phi) \rightarrow \Psi \implies Q^-x \,(\Phi \rightarrow \Psi) \quad x \text{ not free in } \Psi \]

\[ R_3 \quad \Phi \rightarrow (Qy \, \Psi) \implies Qy \,(\Phi \rightarrow \Psi) \quad y \text{ not free in } \Phi \]

\[ R_4 \quad \forall x \, \Phi \land \forall y \, \Psi \implies \forall x \,(\Phi \land \Psi[y/x]) \]

\[ R_5 \quad \exists x \, \Phi \lor \exists y \, \Psi \implies \exists x \,(\Phi \lor \Psi[y/x]) \]

Remarks

- \( Q \in \{\forall, \exists\} \), (Q, Q^-) is (\forall, \exists) or (\exists, \forall) and \( \circ \in \{\land, \lor\} \)

- In general, the PNF of \( \Phi \) is not unique
  (depends, e.g., on rule choice: \( R_1 \) vs \( R_4 \) if both are applicable)

- \( \Phi \) and all of its prenex forms are logically equivalent (why?)
The polynomial hierarchy (PH) again
The polynomial hierarchy (PH) again

Problem complexity determines quantifier prefix of the target QBF
(number of alternations, starting quantifier)
How to handle non-prenex QBFs?
Extend the complexity landscape to arbitrary closed QBFs

- Take the maximal number of quantifier alternations along a path in the syntax tree of a QBF into account

- Almost all QBFs can be translated into equivalent QBFs in PNF without increasing the number of quantifier alternations (Which are the problematic QBFs?)

- Translation procedure is fast but non-deterministic

- Can heavily influence the performance of QBF solvers

Generating CNFs from the matrix

The Tseitin-based algorithm works in three steps

1. Generate a prenex form $\Psi_p: Q_iX_i \cdots Q_kX_k \psi$ of the input QBF $\psi$ with minimal number of quantifier alternations. Then the matrix $\psi$ is purely propositional.

2. Use Tseitin’s translation to transform $\psi$ into CNF.

3. Place the $\exists$ quantifiers for the newly introduced variables $\ell_1, \cdots, \ell_m$ abbreviating $\varphi_1, \ldots, \varphi_m$ “correctly”, e.g.,

   - place all the new $\exists$ at the end of the quantifier prefix, or
   - place $\exists \ell_i$ ($1 \leq i \leq m$) after all quantifiers of those variables which occur in $\varphi_i$.

☞ The use of quantifiers results in an equivalent CNF
Some problem sets and their quantifier structure

- Test sets for non-normal form solvers not easy to find (most test sets are already in PCNF)
- Chosen 3 sets (with increasing complexity of quantifier structure)
  - S1: Encodings for sat in modal logic $K$ (given by Pan and Vardi)
  - S2: Encodings for answer set (AS) correspondence checks
  - S3: Encodings of reasoning with nested counterfactuals
Prenexing strategies and the quantifier structure

- **Prenexing**: "linearize" quantifier dependencies *without* increasing the number of alternations

- We need two strategies here
  - ↑: Place the quantifiers as **outermost** (high) as possible
  - ↓: Place the quantifiers as **innermost** (low) as possible
S2: Encodings for AS correspondence checks

- **Generation of the instances for the $\Pi_2^P$ problems (S22)**
  (from [Janhunen & Oikarinen 2004])
  - QBF generated randomly & translated to LP (similar to S24)
  - Take logic program, remove one clause and check equiv
    (problem has much “propositional” structure!!!)

- Depth of the QBFs: 2, i.e. only 1 quantifier alternation
- Grouped in 8 subsets (QBFs with 10, 12, . . . , 24 variables)
- Number of instances in each subset: 100
- In approx. 50%, the equivalence holds
- Only translation $T$ used
- No specific prenexing strategy (problem is on the 2nd level)

http://www.kr.tuwien.ac.at/research/systems/eq/index.html
Experimental results for the $\Pi_2^P$ problems
A counterfactual (cf) $p > q$ is a conditional query . . .

. . . if we put $p$ to our theory $\mathcal{T}$, can we derive $q$

If $p, q$ are cfs, then $p > q$ is called **nested cf**

Nested counterfactuals span the polynomial hierarchy

Nested counterfactuals can be encoded as QBFs.

Random generation of the problems like before
Statistics for problems from S3

- Grouped in 5 sets (with depth of QBFs from 4 to 8)
- Approximately 60% are valid
- Number of instances per set: 50
- Number of variable: 183, 245, 309, 375, 443
- Number of vars (after PCNF trans): 464, 600, 786, 934, 1132
- Many prenexing strategies tried but . . .
- . . . we show best strategy for each solver in the graphic
Experimental results for nested counterfactuals

Nowadays, many solvers can solve ncf problems (by analyzing “dependencies between variables”)
Experimenta... results for nested counterfactuals

Nowadays, many solvers can solve ncf problems (by analyzing “dependencies between variables”)
Outline

The story so far

Normal form translation again

A resolution calculus for QBFs

Gentzen/sequent systems for QBFs
Why do we need a resolution calculus for QBFs?

- We need a QSAT solver in our rapid implementation approach. Why not Q-resolution?
- Although you will usually not see it, but in nearly every QDPLL solver, there is Q-resolution inside.
- Some QDPLL solvers deliver Q-resolution “refutations” as certificates for unsatisfiability.
- From these proofs, one can generate witness functions (as mentioned earlier).
A resolution calculus for QBFs

Definition (Propositional resolution rule)
Let $C \lor x$ and $D \lor \neg x$ be two clauses, where $C, D$ are disjunctions of literals. With the propositional resolution rule

$$\frac{C \lor x \quad D \lor \neg x}{C \lor D} \quad PRes$$

we derive the resolvent $C \lor D$ from the indicated parent clauses.

Definition (Propositional factor)
Let $C_1 \lor \ell \lor C_2 \lor \ell \lor C_3$ be a clause with possibly empty subclauses $C_1, C_2, C_3$ and let $\ell$ be a literal. With the propositional factor rule

$$\frac{C_1 \lor \ell \lor C_2 \lor \ell \lor C_3}{C_1 \lor \ell \lor C_2 \lor C_3} \quad PFac$$

the factor $C_1 \lor \ell \lor C_2 \lor C_3$ of the indicated clause can be derived.
Definition (Quantification level)
Let $Q$ be a sequence of quantifiers. Associate to each alternation its level as follows. The left-most quantifier block gets level 1, and each alternation increments the level.

Example: $\forall x_1 \forall x_2 \exists y_1 \exists y_2 \exists y_3 \forall x_3 \exists y_4 \varphi$

level 1 level 2 level 3 level 4

Definition ($\forall$ reduction)
Let $C \lor \ell$ be a non-tautological clause, $\ell$ a universal literal and no other literal in $C$ has higher level. Then, with $\forall$ reduction

$$\frac{C \lor \ell}{C} \forall R$$

we can derive $C$ from $C \lor \ell$. 
A resolution calculus for QBFs (cont’d)

**Definition (Q-resolution calculus)**
The Q-resolution calculus consists of the propositional resolution and factor rule and the $\forall$ reduction rule. Resolution operations are only allowed over existential literals and tautological resolvents are deleted immediately.

**Theorem (Kleine Büning, Karpinski, Flögel, Inf. Comput., 1995)**
A PCNF is false iff there is a derivation of the empty clause $\square$ in the Q-resolution calculus.

**Example**
On blackboard
Is the following rule allowed/sound?

**Definition (Possible resolution rule over \( \forall \) variables)**

Let \( C \lor x \) and \( D \lor \neg x \) be two clauses, where \( C, D \) are disjunctions of literals and \( x \) is a universal variable. With the \( \forall \) resolution rule

\[
\begin{array}{c}
C \lor x \\
D \lor \neg x \\
\hline
C \lor D
\end{array}
\]

we derive the resolvent \( C \lor D \) from the indicated parent clauses.
Outline

The story so far

Normal form translation again

A resolution calculus for QBFs

Gentzen/sequent systems for QBFs
Why sequent systems?

- Give a nice way for non-normal form theorem proving (not only for QBFs; also for propositional/FO/non-classical logic)
- Vast amount of proof theoretical knowledge about them
- Easy to implement (as we will see $\implies$ qpro)

Definition (Sequent)

A sequent $S$ is an ordered pair of the form $\Gamma \vdash \Delta$, where $\Gamma$ (antecedent) and $\Delta$ (succedent) are finite multisets of formulas. We write “$\vdash \Delta$” or “$\Gamma \vdash$” whenever $\Gamma$ or $\Delta$ is the empty sequence, respectively.
The propositional rules of a sequent calculus for QBFs

\[
\frac{\Gamma \vdash \Delta}{\phi, \Gamma \vdash \Delta} \quad \text{wl}
\]

\[
\frac{\Gamma_1, \phi, \phi, \Gamma_2 \vdash \Delta}{\Gamma_1, \phi, \Gamma_2 \vdash \Delta} \quad \text{cl}
\]

\[
\frac{\Gamma \vdash \Delta, \phi}{\neg \phi, \Gamma \vdash \Delta} \quad \text{cl}
\]

\[
\frac{\phi, \psi, \Gamma \vdash \Delta}{\phi \land \psi, \Gamma \vdash \Delta} \quad \land l
\]

\[
\frac{\phi, \Gamma \vdash \Delta \quad \psi, \Gamma \vdash \Delta}{\phi \lor \psi, \Gamma \vdash \Delta} \quad \lor l
\]

\[
\frac{\Gamma \vdash \Delta, \phi \quad \psi, \Gamma \vdash \Delta}{\phi \rightarrow \psi, \Gamma \vdash \Delta} \quad \rightarrow l
\]

\[
\frac{\Gamma \vdash \Delta}{\Gamma \vdash \Delta, \phi} \quad \text{wr}
\]

\[
\frac{\Gamma \vdash \Delta_1, \phi, \Delta_2}{\Gamma \vdash \Delta_1, \phi, \Delta_2} \quad \text{cr}
\]

\[
\frac{\phi, \Gamma \vdash \Delta}{\Gamma \vdash \Delta, \neg \phi} \quad \text{cl}
\]

\[
\frac{\Gamma \vdash \Delta, \phi \quad \Gamma \vdash \Delta, \psi}{\Gamma \vdash \Delta, \phi \land \psi} \quad \land r
\]

\[
\frac{\Gamma \vdash \Delta, \phi, \psi}{\Gamma \vdash \Delta, \phi \lor \psi} \quad \lor r
\]

\[
\frac{\phi, \Gamma \vdash \Delta, \psi}{\Gamma \vdash \Delta, \phi \rightarrow \psi} \quad \rightarrow r
\]
The axioms and quantifier rules for the calculus

The axioms: \( \Phi \vdash \Phi \ \text{Ax} \quad \bot \vdash \bot \quad \top \vdash \top \quad \top \vdash \top \)

Some possible quantifier rules:

\[
\frac{\Gamma \vdash \Delta, \Phi\{p/q\}}{\Gamma \vdash \Delta, \forall p \Phi} \quad \text{\(\forall r_e\)}
\]

\[
\frac{\Phi\{p/\psi\}, \Gamma \vdash \Delta}{\forall p \Phi, \Gamma \vdash \Delta} \quad \text{\(\forall l_f\)}
\]

\[
\frac{\Phi\{p/\top\}, \Phi\{p/\bot\}, \Gamma \vdash \Delta}{\forall p \Phi, \Gamma \vdash \Delta} \quad \text{\(\forall l_s\)}
\]

\[
\frac{\Gamma \vdash \Delta, \Phi\{p/\top\} \land \Phi\{p/\bot\}}{\Gamma \vdash \Delta, \forall p \Phi} \quad \text{\(\forall r_s\)}
\]

\[
\frac{\Phi\{p/\top\} \lor \Phi\{p/\bot\}, \Gamma \vdash \Delta}{\exists p \Phi, \Gamma \vdash \Delta} \quad \text{\(\exists l_s\)}
\]

\[
\frac{\Phi\{p/\psi\}, \Gamma \vdash \Delta}{\exists p \Phi, \Gamma \vdash \Delta} \quad \text{\(\exists l_e\)}
\]

\[
\frac{\Gamma \vdash \Delta, \Phi\{p/\psi\}}{\exists p \Phi, \Gamma \vdash \Delta} \quad \text{\(\exists r_f\)}
\]

\[
\frac{\Gamma \vdash \Delta, \Phi\{p/\top\}, \Phi\{p/\bot\}}{\exists p \Phi, \Gamma \vdash \Delta} \quad \text{\(\exists r_s\)}
\]

\[
\frac{\Gamma \vdash \Delta, \Phi\{p/\top\} \land \Phi\{p/\bot\}}{\exists p \Phi, \Gamma \vdash \Delta} \quad \text{\(\exists l_s\)}
\]

\[q \text{ does not occur as a free variable in the conclusion of } \forall r_e / \exists l_e\]
The basic algorithm

- Based on DPLL (successful in SAT-/QBF-solving in (P)CNF)
- Relatively simple extension for nonprenex QBFs in NNF (implementation follows the semantics using $s$ quantifier rules)

```cpp
BOOLEAN split(QBF $\phi$ in NNF) {
    switch (simplify ($\phi$)): /* simplify works inside $\phi$ */
        case $\top$: return True;
        case $\bot$: return False;
        case ($\phi_1 \lor \phi_2$): return (split($\phi_1$) || split($\phi_2$));
        case ($\phi_1 \land \phi_2$): return (split($\phi_1$) && split($\phi_2$));
        case ($\forall X \psi$): select $x \in X$;
            if $Q = \exists$ return (split($\exists X \psi[x/\bot]$) || split($\exists X \psi[x/\top]$));
            if $Q = \forall$ return (split($\forall X \psi[x/\bot]$) && split($\forall X \psi[x/\top]$));
    }
```
Simplifying formulas

**simplify(\(\phi\)):** returns \(\phi'\) simplified wrt some equivalences:

(a) \(\neg \top \Rightarrow \bot; \quad \neg \bot \Rightarrow \top;\)

(b) \(\top \land \phi \Rightarrow \phi; \quad \bot \land \phi \Rightarrow \bot; \quad \top \lor \phi \Rightarrow \top; \quad \bot \lor \phi \Rightarrow \phi;\)

(c) \((Qx \phi) \Rightarrow \phi, \text{ if } Q \in \{\forall, \exists\}, \text{ and } x \text{ does not occur in } \phi;\)

(d) \(\forall x(\phi \land \psi) \Rightarrow (\forall x \phi) \land (\forall x \psi);\)

(e) \(\forall x(\phi \lor \psi) \Rightarrow (\forall x \phi) \lor \psi, \text{ whenever } x \text{ does not occur in } \psi;\)

(f) \(\exists x(\phi \lor \psi) \Rightarrow (\exists x \phi) \lor (\exists x \psi);\)

(g) \(\exists x(\phi \land \psi) \Rightarrow (\exists x \phi) \land \psi, \text{ whenever } x \text{ does not occur in } \psi.\)

Rewritings (d)–(g) are known as **miniscoping**.
Additional mechanisms

- Basic procedure clearly not sufficient for competitive solver
- Desirable extension: generalization of pruning techniques
  - Unit literal elimination
  - Pure literal elimination
  - Dependency-directed backtracking (works for true and false subproblems)
  - Learning

\(\Rightarrow\) split looks like an implementation of a sequent calculus

\(\Rightarrow\) Extensions of split formalized as a sequent calculus (for NNF)
The Logical Rules of the Sequent Calculus $\text{GQBF}_1$

\[
\begin{align*}
\frac{\vdash \phi}{\vdash \phi \lor \psi} \quad \text{(∨')} \\
\frac{\vdash \psi}{\vdash \phi \lor \psi} \quad \text{(∨'')} \\
\frac{\vdash \phi \land \psi}{\vdash \phi \lor \psi} \quad \text{(^)}
\end{align*}
\[
\begin{align*}
\frac{\vdash \phi[x/\bot]}{\vdash \exists x \phi} \quad \text{(^\prime)} \\
\frac{\vdash \phi[x/\top]}{\vdash \exists x \phi} \quad \text{(^\prime')} \\
\frac{\vdash \phi[x/\bot]} {\vdash \phi[x/\top]} \quad \text{(^\prime\prime)} \\

\frac{\vdash \forall x \phi}{\vdash \exists x \phi} \quad \text{(∀)}
\end{align*}
\]

- The logical rules (l-rules) simulate the basic algorithm
- Sequents consist of one formula only
- Axioms of $\text{GQBF}_1$ are $\vdash \top$ and $\vdash \neg \bot$

- Use proofs in $\text{GQBF}_1$ to prove properties of techniques like DDB more elegantly
- Are such proofs in $\text{GQBF}_1$ useful for applications?
Simplification Rules

- **GQBF** \(_1\) extended by **simplification rules** like

\[
\frac{\vdash \phi(T)}{\vdash \phi(T \lor \psi)} \quad (S3a) \quad \frac{\vdash \phi(\psi)}{\vdash \phi(\bot \lor \psi)} \quad (S3b)
\]

\[
\frac{\vdash \phi(\psi)}{\vdash \phi(Qx \psi)} \quad (S4) \text{ no occurrences of } x \text{ in } \psi
\]

- Very important that they work **inside** formulas (circumventing the usual decomposition strategy)

- Allow for short proofs for (simple) formulas like

\[
\exists x_n \forall y_n \cdots \exists x_1 \forall y_1 (x_n \lor y_n \lor \cdots \lor x_1 \lor y_1)
\]

- ... which have only long proofs in GQBF\(_1\)