Algorithms for Computational Logic

Summary

Contents

Chapter 1

SAT and Modeling with SAT

1.1 Cardinality Constraints

In order to handle cardinality constraints we have two options: encode the cardinality constraints to CNF and use a SAT solver, or use a pseudo boolean (PB) solver.

1.1.1 AtMost1

- $\sum_{j=1}^{n} x_j = 1$ can be encoded with $\left(\sum_{j=1}^{n} x_j \leq 1\right) \wedge \left(\sum_{j=1}^{n} x_j \geq 1\right)$
- $\sum_{j=1}^{n} x_j \geq 1$ can be encoded with $(x_1 \vee x_2 \vee ... \vee x_n)$
- $\sum_{j=1}^{n} x_j \leq 1$ can be encoded with:
  - Pairwise encoding
  - Sequential counter encoding
  - Bitwise encoding

Sequential Counter

In order to realize this encoding, we need to add new variables s_i for the fact "there is a 1 on some position $1..i$":

$$s_i \text{ is true if } \sum_{j=1}^{i} x_j \geq 1$$

Encoding $\sum_{j=1}^{n} x_j \leq 1$ with sequential counter:

$$\begin{aligned}
&(\neg x_1 \vee s_1) \wedge \\
&(\neg x_i \vee s_i), i \in 2..n-1 \wedge \\
&(\neg s_{i-1} \vee s_i), i \in 2..n-1 \wedge \\
&(\neg x_i \vee \neg s_{i-1}), i \in 2..n
\end{aligned}$$

If $x_j = 1$, then all s_i variables are assigned and all other x variables must take value 0. There are $\mathcal{O}(n)$ clauses and $\mathcal{O}(n)$ auxiliary variables.

Bitwise Encoding

In bitwise encoding, we represent the constraint $\sum_{j=1}^{n} x_j \leq 1$ by encoding the index of the potential true variable in binary. For this, we add new auxiliary variables:

$$v_0, ...v_r - 1;\ r = \lceil \log n \rceil (\text{with } n > 1)$$

Each variable x_j is assigned a unique binary number that represents its index. Then, for each variable x_j with binary index representation i, we create clauses that enforce the condition:

- If $x_j = 1$, assignment to v_i variables must encode $j - 1$, and all other x variables must take value 0
- If all $x_j = 0$, any assignment to v_i variables is consistent

For example, $x_1 + x_2 + x_3 \leq 1$:

	$j-1$	$v_1 v_0$
x_1	0	00
x_2	1	01
x_3	2	10

$$(\neg x_1 \vee \neg v_1) \wedge (\neg x_1 \vee \neg v_0)$$
$$(\neg x_2 \vee \neg v_1) \wedge (\neg x_2 \vee v_0)$$
$$(\neg x_3 \vee v_1) \wedge (\neg x_3 \vee \neg v_0)$$

There are $\mathcal{O}(n \log n)$ clauses and $\mathcal{O}(\log n)$ auxiliary variables

1.1.2 General Cardinality Constraints

Constraints of the form $\sum_{j=1}^{n} x_j \leq k$ or $\sum_{j=1}^{n} x_j \geq k$ can be added with:

- Sequential Counters
- BDDs
- Sorting Networks
- Cardinality Networks
- Totalizer

Sequential Counter Encoding

For each variable x_i, create k additional variables $s_{i,j}$ that are used as counters:

- $s_{i,j} = 1$ if at least j variables $\{x_1 ... x_i\}$ are assigned value 1
- $s_{i,j} = 0$ if at most $j - 1$ variables $\{x_1 ... x_i\}$ are assigned value 1

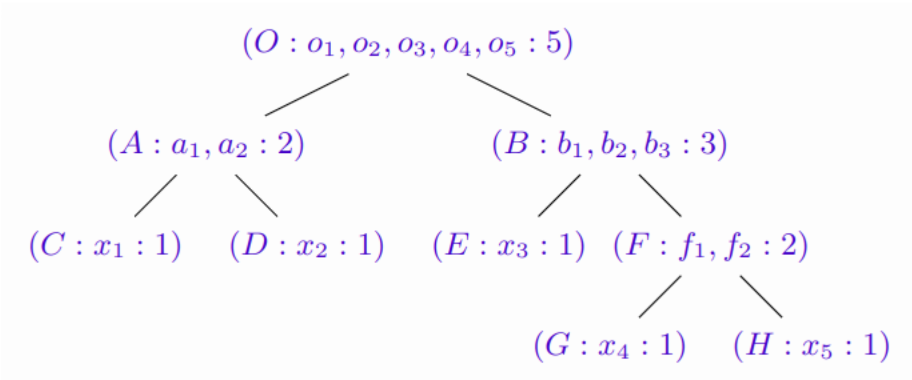
Encoding:

$$
\begin{array}{ll}
(\neg x_1 \vee s_{1,1}) & \\
(\neg s_{1,j}), & \forall j : 1 < j \leq k \\
 & \\
(\neg x_i \vee s_{i,1}), & \forall i : 1 < i < n \\
(\neg s_{i-1,1} \vee s_{i,1}), & \forall i : 1 < i < n \\
 & \\
(\neg s_{i-1,j} \vee s_{i,j}) & \forall i,j : 1 < i < n, 1 < j \leq k \\
(\neg x_i \vee \neg s_{i-1,j-1} \vee s_{i,j}) & \forall i,j : 1 < i < n, 1 < j \leq k \\
 & \\
(\neg x_i \vee \neg s_{i-1,k}) & \forall i : 1 < i \leq n
\end{array}
$$

Totalizer Encoding

In this encoding we count in unary how many of the n variables $(x_1 ... x_n)$ are assigned to 1. It can be visualized as a tree:

- Each node is $(name : variable : sum)$
- Root node has the output variables $(o_1 ... o_n)$ that count how many variables are assigned to 1
- Literals are at the leaves
- Each node counts in unary how many leaves are assigned to 1 in its subtree
- Example: if $b_2 = 1$, then at least 2 of the leaves (x_3, x_4, x_5) are assigned to 1



To encode $x_1 + x_2 + x_3 + x_4 + x_5 \leq 3$ just set $o_4 = 0$ and $o_5 = 0$.
Encoding:

$$
\bigwedge_{\substack{0 \leq \alpha \leq n_2 \\ 0 \leq \beta \leq n_3 \\ 0 \leq \sigma \leq n_1 \\ \alpha + \beta = \sigma}} \neg q_\alpha \vee \neg r_\beta \vee p_\sigma \quad \text{where, } p_0 = q_0 = r_0 = 1
$$

There are $\mathcal{O}(n \log n)$ new variables and $\mathcal{O}(n^2)$ new clauses

Chapter 2

Optimization problems and SAT-Based Problem Solving

Chapter 3

Satisfiability Modulo Theories

Chapter 4

Answer Set Programming