

FRANCESCO ANTICI

SAT CONSTRAINTS EMBEDDINGS

CARDINALITY CONSTRAINTS

- ▶ In most of the exercises we solved we found a cardinality constraint.
- ▶ Cardinality constraints are the ones in the form:

$$p_1 + \dots + p_n \stackrel{\leq}{=} k$$

CARDINALITY CONSTRAINTS

- ▶ In sat these constraints are represented by:
 - ▶ *at_least_one*, *at_most_one* and *exactly_one*;
 - ▶ *at_least_k*, *at_most_k* and *exactly_k*.
- ▶ The encodings we presented can be very inefficient with big instances, so we need to find better ones.

CARDINALITY CONSTRAINTS

- Keeping in mind that:

$$at_least_k(V, k) \equiv at_most_k(\{ \neg v \mid v \in V \}, |V| - k)$$

$$exactly_k(V) \equiv at_most_k(V) \wedge at_least_k(V)$$

- We are going to focus just on the *at_most_k* constraint.

AT MOST ONE-PAIRWISE ENCODING

- ▶ The pairwise(or naive) encoding of the *at_most_one* constraint is:

$$\bigwedge_{1 \leq i < |V|} \bigwedge_{i+1 \leq j \leq |V|} \neg(V_i \wedge V_j)$$

- ▶ This encoding doesn't require the addition of any new variables, but it encodes $O(n^2)$ clauses, with $n = |V|$.

AT MOST ONE-SEQUENTIAL ENCODING

- ▶ The sequential encoding of the *at_most_one*(V) constraint consists of using $n - 1$ variables s_i to keep track of which V_i is true, it is encoded as follows:

$$(\neg V_1 \vee s_1) \wedge (\neg V_n \vee \neg s_{n-1}) \wedge \bigwedge_{1 < i < n} ((\neg V_i \vee s_i) \wedge (\neg s_{i-1} \vee s_i) \wedge (\neg V_i \vee \neg s_{i-1}))$$

- ▶ This encoding produces $3n - 4(O(n))$ clauses, with $n = |V|$.

AT MOST ONE-BITWISE ENCODING

- ▶ The bitwise encoding of the *at_most_one*(*V*) constraint consists of using $\log_2(n)$ new variables $r_1, \dots, r_{\log_2(n)}$ to represent the binary encoding of the index of the variable which is true, so it is encoded like:

$$\bigwedge_{1 \leq i < n} \bigwedge_{1 \leq j < \log_2(n)} \neg V_i \vee r_{i,j}$$

- ▶ This encoding produces $n \log_2(n)$ clauses, with $n = |V|$.

AT MOST ONE-HEULE ENCODING

- ▶ The Heule encoding is another linear version of the $at_most_one(V)$ constraint applicable for $n > 4$, which consists of splitting the pairwise encoding in two parts, adding an auxiliary variable y :

$$at_most_one(V_1, \dots, V_3, y) \wedge at_most_one(\neg y, V_4, \dots, V_n)$$

- ▶ This encoding require the addition of 1 new variable, but it encodes $O(n)$ clauses, with $n = |V|$.

AT MOST K-PAIRWISE ENCODING

- ▶ The pairwise(or naive) encoding of the *at_most_k* constraint is:

$$at_most_k(V, k) \equiv \bigwedge_{X \subseteq V} \bigvee_{v \in X} \neg v$$

- ▶ This encoding doesn't require the addition of any new variables, but it encodes $\binom{n}{k+1}$ clauses of length $k+1$, with $n = |V|$.
- ▶ In the worst case, it can amount to a $O(2^n / \sqrt{n/2})$

AT MOST K-SEQUENTIAL ENCODING

- ▶ The sequential encoding of the $at_most_k(V, k)$ constraint consists of using $(n - 1) * k$ variables s_i to keep track of which V_i is true, it is encoded as follows:

$$\left. \begin{array}{l}
 (\neg x_1 \vee s_{1,1}) \\
 (\neg s_{1,j}) \quad \text{for } 1 < j \leq k \\
 (\neg x_i \vee s_{i,1}) \\
 (\neg s_{i-1,1} \vee s_{i,1}) \\
 (\neg x_i \vee \neg s_{i-1,j-1} \vee s_{i,j}) \\
 (\neg s_{i-1,j} \vee s_{i,j}) \\
 (\neg x_i \vee \neg s_{i-1,k}) \\
 (\neg x_n \vee \neg s_{n-1,k})
 \end{array} \right\} \text{ for } 1 < j \leq k \quad \left. \vphantom{\begin{array}{l} (\neg x_1 \vee s_{1,1}) \\ (\neg s_{1,j}) \\ (\neg x_i \vee s_{i,1}) \\ (\neg s_{i-1,1} \vee s_{i,1}) \\ (\neg x_i \vee \neg s_{i-1,j-1} \vee s_{i,j}) \\ (\neg s_{i-1,j} \vee s_{i,j}) \\ (\neg x_i \vee \neg s_{i-1,k}) \\ (\neg x_n \vee \neg s_{n-1,k}) \end{array}} \right\} \text{ for } 1 < i < n$$

- ▶ This encoding needs $2nk + n - 3k - 1$ clauses, with $n = |V|$.