# Appendix for:
# I Always Told My Mom That Order Is Overrated: Unordered HTN Planning is in PSPACE and Models Problems Beyond STRIPS

**Pascal Lauer**[1,2], **Yifan Zhang**[1], **Patrik Haslum**[1], **Pascal Bercher**[1]

[1]School of Computing, The Australian National University, Canberra, Australia
[2]Saarland Informatics Campus, Saarland University, Saarbrücken, Germany
firstname.lastname@anu.edu.au

## A Addendum Lemma 4.4

### A.1 Conversion to Linear Constraints

We first describe how constraints as described in Lemma 4.4 can be converted to linear constraints. Let $C$ denote the set of all constants occurring in initial state, goal, and operator effects. Further, let $b^* = \prod_{\frac{a}{b} \in C} b$ be a constant that cancels all denominators in the planning problem. The construction in Def. 2.4 makes it clear that multiplying every constant in the problem does not affect the plans in the problem. The multiplication causes a polynomial size increase of the planning problem.

So, w.l.o.g., we describe the conversion for numerical planning problems where all constants occuring in initial state, goal, and effects are integers. $BC^+$ numerical planning captures the goal constraints

$$x_1 - c_0 \; relop \; 0 \text{ and } x_1 - x_2 \; relop \; 0$$

for $relop \in \{=, <, \leq, \geq, >, \neq\}$ with variables $x_1, x_2$ and constant $c_0 \in \mathbb{Z}$. Solutions to the constraints are assignments $\beta : \{x_1, x_2\} \to \mathbb{N}$ so that the constraint is fulfilled. The restriction of solutions to $\mathbb{N}$, instead of $\mathbb{Q}$, is valid as: (1) By considering $BC^+$ numerical planning, it is okay to restrict solutions to $\mathbb{Q}_0^+$, as the additional goal conditions enforce all solution values to be $\geq 0$. (2) The assumption to use integers means that $\mathbb{N}$ is the actual considered subset of $\mathbb{Q}_0^+$. We show the conversion for the more general form:

$$c_1 x_1 + ... + c_n x_n \; relop \; c_0$$

Where $x_1, ..., x_n$ are variables, $c_0, ..., c_n \in \mathbb{Z}$ constants. Solutions to the constraints are assignments $\beta : \{x_1, ..., x_n\} \to \mathbb{N}$ so that the constraint is fulfilled. This describes the conversion for both conditions. Now, we describe each conversion depending on the choice of $relop$.

**(1) Case "=":** The is the condition:

$$c_1 x_1 + ... + c_n x_n = c_0$$

And matches exactly the constraint expressed by linear constraint $\mathbb{L} = (M, c_0)$ with a one row matrix

$$M = (c_1, ..., c_n)$$

**(2) Case "<":** This is the condition:

$$\begin{aligned} & c_1 x_1 + ... + c_n x_n && \leq c_0 \\ \equiv\ & c_1 x_1 + ... + c_n x_n + (-1) x_{n+1} && = c_0 \end{aligned}$$

The equivalence holds as all solution values for linear constraints are, by our definition, in $\mathbb{N}$, so this guarantees $x_{n+1} \geq 0$. This is a standard construction. It is common to call $x_{n+1}$ slack variable.

**(3) Case "≤":** This is the condition:

$$\begin{aligned} & c_1 x_1 + ... + c_n x_n < c_0 \\ \equiv\ & c_1 x_1 + ... + c_n x_n \leq (c_0 - 1) \end{aligned}$$

The equivalence holds as all solutions and constants are integers.

**(4) Case "≥":** This is the condition:

$$\begin{aligned} & c_1 x_1 + ... + c_n x_n \geq c_0 \\ \equiv\ & -c_1 x_1 + ... + -c_n x_n \leq -c_0 \end{aligned}$$

So, we can just reuse the result from (3).

**(5) Case ">":** This is the condition:

$$\begin{aligned} & c_1 x_1 + ... + c_n x_n > c_0 \\ \equiv\ & -c_1 x_1 + ... + -c_n x_n < -c_0 \end{aligned}$$

So, we can just reuse the result from (2).

**(6) Case "≠":** This is the condition:

$$c_1 x_1 + ... + c_n x_n \neq c_0$$

which must match

$$\begin{aligned} & c_1 x_1 + ... + c_n x_n < c_0 \\ \text{or } & c_1 x_1 + ... + c_n x_n > c_0 \end{aligned}$$

So, as we only need to show there exists an encoding, we know there is one with the construction from (5) or (2).

**Combining Multiple Constraints** All of this explains the conversion for one constraint at a time. Multiple constraints can be converted by combining the matrices in a way so that all introduced slack variables are different.

## A.2 Construction of $p'$

A.1 shows how to convert the the constraints outlined in Lemma 4.4 into linear constraints $\mathbb{L} = (M, v)$ with matrix $M \in \mathbb{Z}^{m \times n}$ and a vector $v \in \mathbb{Z}^m$. We now make use of the results by Papadimitriou (1981), to determine the bound on solution values captured via the polynomial $p'$. Papadimitriou (1981) states that if $\text{sol}(\mathbb{L}) \neq \emptyset$, then there is a solution $x \in \text{sol}(\mathbb{L})$ so that

$$\forall i \in \{1, ..., m\} : 0 \leq x_i \leq n(m \cdot \max(\mathbb{L}))^{2m+1}$$

Where $\max(\mathbb{L})$ denotes the highest constant occuring in $M, v$.

**Bounding $m, n$:** From the bounds determined in the proof of Lemma 4.4 and the transformation in A.1, it is easy to determine fitting bounds on $n, m$:

- Number of solution variables $n$: The equality constraints outlined in Lemma 4.4 contain $\leq |V_N| + 2^{|V_P| \cdot |Ops|}$ variables. The transformation to linear constraints introduces at most one (slack) variable per goal constraint during the transformation to linear constraints. This bounds $n \leq |V_N| + 2^{|V_P| \cdot |Ops|} + |Goal|$.
- Number of rows $m$: We do not increase the number of constraints, which bounds $m \leq |V_N| + |Goal|$.

**Bounding $\max(\mathbb{L})$ w.r.t. $b^*, C$:** Additionally we need to upper-bound $\max(\mathbb{L})$. Let $C$ be defined as in A.1 as the set of all constant ocurring in $\Pi$. The maximum number occuring in $C$ is denoted by $\max(C)$. Let $b^*$ be the scaling factor from A.1. We first bound $\max(\mathbb{L})$ w.r.t. $C$ and $b^*$.

- In the equality constraints determining variable values expressed in Lemma 4.4, there are at most $2^{|V_P|} + 1$ constants not associated with a variable, which would be added up to a single constant bounded by $\leq (2^{|V_P|} + 1) \cdot b^* \cdot \max(C)$.
- In the equality constraints determining variable values expressed in Lemma 4.4, each variable $x_{c'_i}$ is associated with a unique constant $\Delta_v(c'_i)$ created from adding up the operators costs in a simple propositional cycle. This bounds such constants by $\leq 2^{|V_P|} \cdot b^* \cdot \max(C)$.
- To write the goal conditions into linear form one adds up at most $||\Pi||$ constants with value at most $b^* \cdot \max(C)$. I.e., the maximum constant generated from a goal condition is $\leq ||\Pi|| \cdot b^* \cdot \max(C)$.

This bounds the constants in all constraints we encode, and so:

$$\max(\mathbb{L}) \leq b^* \cdot \max(C) \cdot (||\Pi|| + 2^{|V_P|} + 2^{|V_P|} + 1)$$

**Bounding $b^*, C$:** As $C$ is the set of constants occuring in $\Pi$, we get:

$$\max(C) \leq 2^{||\Pi||}$$

As there are at most $||\Pi||$ elements in $C$, it holds:

$$b^* \leq |C| \cdot \max(C) \leq ||\Pi|| \cdot 2^{||\Pi||}$$

**Constructing $p'$:** These bounds already suffice to see that a $p'$ as described in Lemma 4.4 exists. For completeness, we further loosen the bound to construct a term that could be used to construct $p'$ directly. We use the bound determined for $m \leq |V_N| + |Goal|$ and shape up the others to one matching the pattern $2^{<polynomial>}$:

$$n \leq |V_N| + 2^{|V_P| \cdot |Ops|} + |Goal| \leq 2^{|V_N| + |V_P| \cdot |Ops| + |Goal|}$$

$$b^* \cdot \max(C) \leq 2^{||\Pi||} \cdot ||\Pi|| \cdot 2^{||\Pi||} \leq 2^{3||\Pi||}$$

$$\max(\mathbb{L}) \leq 2^{3||\Pi||} \cdot (||\Pi|| + 2^{|V_P|} + 2^{|V_P|} + 1) \leq 2^{4||\Pi|| + 3|V_P|}$$

Then we simply input these bounds into the observation by Papadimitriou (1981):

$$\begin{aligned}
& n(m \cdot \max(\mathbb{L}))^{2m+1} \\
\leq\ & 2^{|V_N| + |V_P| \cdot |Ops| + |Goal|} \\
& \cdot ((|V_N| + |Goal|) \cdot 2^{4||\Pi|| + 3|V_P|})^{2 \cdot (|V_N| + |Goal|) + 1} \\
\leq\ & 2^{|V_N| + |V_P| \cdot |Ops| + |Goal|} \\
& \cdot (2^{|V_N| + |Goal| + 4||\Pi|| + 3|V_P|})^{2 \cdot (|V_N| + |Goal|) + 1} \\
\leq\ & 2^{|V_N| + |V_P| \cdot |Ops| + |Goal| + (|V_N| + |Goal| + 4||\Pi|| + 3|V_P|) \cdot (2 \cdot (|V_N| + |Goal|) + 1)}
\end{aligned}$$

*In the inequalities we assume that all values $|V_N|, ||\Pi||, |V_P|, |Goal|, |Ops| > 0$. This can be achieved by a simple task transformation.*

# B Addendum Proposition 4.7: Converting a Presburger Formula to Linear Constraints

We are given a Presburger formula over variables $X = \{x_1, ..., x_n\}$ of shape

$$\exists x_1 : ... \exists x_n : \bigwedge_{i=1}^{m} \psi_i$$

where all $\psi_i$ are quantifier-free and atomic formulas of shape $t_1 = t_2$ or $t_1 < t_2$. We encode the formula into a linear constraint $\mathbb{L} = (M, v)$.

**Shape of $\mathbb{L}$** Matrix $M$ has $m$ rows and $2n$ columns. Each row $i \in \{1, .., m\}$, together with the $i$-th value of $v$, forms a constraints of shape $\sum_{i=1}^{2n} c_i \cdot x_i \leq c_0$, corresponding to $\psi_i$. This allows matching each row with constants $c_1, ..., c_n \in \mathbb{Q}$ to variable values of $x_1, ..., x_n$, and add a slack variable per row, if needed.

**Conversion for "="** Given that Presburger formulas are interpreted over $(\mathbb{N}, \leq, +)$ we can use standard algebraic rewriting to bring $t_1 = t_2$ to shape $\sum_{i=1}^{n} c_i^* \cdot x_i = c_0^*$. Here $c_i^* \cdot x_i$ represents $c_i^* \in \mathbb{N}$ additions of $x_i$ and $c_0^*$ is a sum of constants $0, 1$.

**Conversion for "<"** For $t_1 < t_2$ we use the same rewriting, to generate the shape $\sum_{i=1}^{n} c_i^* \cdot x_i < c_0^*$. Which like in A.1 (3) we can represent as equation:

$$(-1 \cdot x_{n+i}) + \sum_{i=1}^{n} c_i \cdot x_i = (c_0 - 1)$$

Where $x_{n+i}$ is the unique slack variable for $\psi_i$.