The *bit-vector* Constraint

Lucas Bordeaux & Youssef Hamadi {LUCASB/YOUSSEFH}@MICROSOFT.COM
Microsoft Research
7 J J Thomson Avenue
CB3 0FB Cambridge
UNITED KINGDOM
Claude-Guy Quimper QUIMPER@ALUMNI.UWATERLOO.CA
4895 Berri # 613
Montréal QC H2J 4A3
CANADA

Abstract

Some applications require to reason on particular bits of an integer value, and to express the fact that *"the number X is encoded in binary by the vector of Boolean variables $[x_n, \ldots, x_0]$"*. The natural way to encode this is using a linear constraint. We show that bound propagation on this constraint has intriguing properties: it is *complete* in the sense that the bounds of the variable $x_i,\ i \in 0 \ldots n$ are tightly reduced; on the other hand, the interval of values for X is in general not optimally reduced: it can be up to twice as large as the optimal. We show that a simple mechanism allows the reasoning to be complete on X.

Keywords: Constraints, Interval Propagation.

1. The bit-vector Constraint

A number of applications, essentially in verification, require to express constraints on particular bits of integer variables. The connection between the integer value X and the bit representation $\langle x_n, \ldots, x_0 \rangle$ is easily encoded by the constraint:

$$X \ = \sum_{i=0\ldots n} 2^i x_i \tag{1}$$

X is an integer variable ranging over $[0, 2^{n+1} - 1]$ and the x_is range over $\{0, 1\}$.

More generally, similar encodings can be used to represent *tuples* of values: if $x_n, \ldots, x_0$ are variables that all range over the same (small) domain $[0 \ldots d-1]$, the tuple $\langle x_n, \ldots, x_0 \rangle$ can be represented by an integer variable X; the constraint is then $X = \sum_{i=0\ldots n} d^i x_i$. For simplicity we shall present the results of this paper for the case of binary domains, but they easily generalize to any arbitrary basis.

Since the variable X can have a large domain (exponential in the number of bits or components), we shall typically represent the set possible values for a variable using an interval representation. The question is then how to achieve the best interval propagation possible for this constraint: is it the case that a basic interval propagation, directly applied to the linear constraint, will make all the correct interval reductions; or can we design

©2006 Bordeaux, Hamadi, Quimper.

an improved, specialized propagator for bit-vect? This note answers these questions. Our results are the following:

1. If we use the linear encoding and perform bound propagation on it, the bounds for the pseudo-Boolean variables x_i ($i \in 0 \dots n$) are reduced optimally;

2. On the other the interval of values for X is not reduced optimally: it can be up to twice as large as the interval that is reduced optimally;

3. We show that a simple algorithm allows to reduce X optimally.

The notion of "reduced optimally" is made clear in the following Section. The first result is presented more formally in Section 3; the second result in Section 4, and the third result in Section 5. We start by presenting basic material related to interval propagation.

2. Basic Material

Given a variable $y \in \{x_0, \dots, x_n, X\}$, we denote by y^- and y^+ its lower and upper bounds. Let:

$$\begin{cases} \sigma^- &= \sum_{i=0\dots n} 2^i x_i^- \\ \sigma^+ &= \sum_{i=0\dots n} 2^i x_i^+ \end{cases}$$

represent the bounds of the sum $\sum_{i=0\dots n} 2^i x_i$. Given a tuple of values $t = \langle a_n, \dots, a_0 \rangle$ we denote by $eval(t)$ the value $\sum_{i=0\dots n} 2^i a_i$.

We give brief a reminder on the notions of *bound propagation* and *bound consistency.* Bound propagation on discrete domains was introduced by Davis (1987) and Cleary (1987); more recent references on this topic are Yuanlin and Yap (2000); Trick (2001); Harvey and Stuckey (2003).

2.1 Propagation

Bound propagation works by considering each variable in turn, and checking whether its lower/upper bounds can be tightened without loosing any solution. For variable X, propagation will make sure that all the values of its range that are inferior to σ^- or superior to σ^+ are discarded. For the Boolean variables x_i, the following reasoning will be applied: *if we fix x_i to 0 and all the other variables x_j ($j \neq i$) to their upper bound then we have to obtain something at least as large as x^-.* Otherwise value 0 can clearly be discarded for x_i. Symmetrically, *if we fix x_i to 1 and all the other variables x_j ($j \neq i$) to their lower bound then we have to obtain something smaller than x^+.* Otherwise value 1 can clearly be discarded for x_i. Propagation repeats these rules for each variable until no bound reduction is possible anymore. The ranges are said to be *stable under propagation* iff we have:

$$\forall i \in 0 \dots n. \begin{cases} \sigma^- \ \leq \ X^- \ \leq \ \sigma^+ + 2^i(x_i^- - x_i^+) \\ \sigma^+ \ \geq \ X^+ \ \geq \ \sigma^- + 2^i(x_i^+ - x_i^-) \end{cases} \tag{2}$$

2.2 Bound Consistency

The variables are said to be "bound-consistent" when their bounds have been optimally reduced, *i.e.,* some solutions would be lost if we reduced these bounds further. More formally, a value v of a variable y is *consistent* if there is a solution that assigns v to y. The linear equation is *bound-consistent* if each bound (y^- and y^+, for $y \in \{x_0 \dots x_n, X\}$) of every variable is consistent.

Let us insist that, in general, the intervals obtained after propagation are *not* bound-consistent (which is why we use the term "stable under propagation" to describe such intervals, instead of a term that would use the word "consistency").

3. The bit variables are reduced optimally

Our first result states that propagation reduces the variables $x_i,\ i \in 0 \dots n$ in an optimal way:

Proposition 1 *Given a constraint of the form (1); if the bounds are stable under propagation then each variable $x_i,\ i \in 0 \dots n$ is bound-consistent.*

This proves, in particular, that interval propagation is *complete* in the sense that if non-empty intervals are computed, we have the guarantee to have a solution within these ranges.

To prove this result, we suppose the ranges are stable under propagation. We prove that the bounds of every x_i have a support. The idea is that propagation will only be able to instantiate some of the variables of highest weight. Let l be the index of the non-instantiated variable of highest weight. For $i > l$ we denote by c_i the constant $x_i^+ = x_i^-$. The claim is proved in three steps:

1. We prove that the upper bound of x_l has a support.
2. We prove that the lower bound of x_l has a support.
3. We prove that both bounds of every x_i, $i \neq l$, have a support.

3.1 The upper bound of x_l has a support

We prove by contradiction that $x_l = 1$ has a support. We define $C = \sum_{i=l+1\dots n} 2^i c_i$. Since the bounds are stable under propagation we have from Eq. 2:

$$x^- - C \ \leq \sum_{i=0\dots l-1} 2^i x_i^+ \qquad (3)$$

$$2^l + \sum_{i=0\dots l-1} 2^i x_i^- \ \leq \ x^+ - C \qquad (4)$$

Now if we suppose that $x_l = 1$ has no support, this means that a lexicographic iteration from $\alpha = \langle c_n, \dots, c_{l+1}, 1, x_{l-1}^-, \dots, x_0^- \rangle$ to $\omega = \langle c_n, \dots, c_{l+1},\ 1, x_{l-1}^+, \dots, x_0^+ \rangle$ never goes through a tuple t satisfying $eval(t) \in [x^-, x^+]$. Because $eval(\alpha) \leq x^-$ and $eval(\omega) \geq x^+$, at some point we have a tuple t that is such that $eval(t) < x^-$ and whose lexicographical successor t' is such that $eval(t') > x^+$.

- t can be written as:

$$\langle c_n, \ldots, c_{l+1}, 1, c_{l-1}, \ldots, c_{j+1}, 0, x^+_{j-1}, \ldots, x^+_0 \rangle$$

- and the next tuple t' as:

$$\langle c_n, \ldots, c_{l+1}, 1, c_{l-1}, \ldots, c_{j+1}, 1, x^-_{j-1}, \ldots, x^-_0 \rangle$$

for a particular choice of constants $c_{j+1} \ldots c_{l-1}$. We therefore translate the fact that $eval(t) < x^-$ and that $eval(t') > x^+$:

$$2^l + \sum_{i=j+1 \ldots l-1} 2^i c_i + 0 + \sum_{i=0 \ldots j-1} 2^i x^+_i \ < \ x^- - C \tag{5}$$

$$x^+ - C \ < \ 2^l + \sum_{i=j+1 \ldots l-1} 2^i c_i + 2^j + \sum_{i=0 \ldots j-1} 2^i x^-_i \tag{6}$$

By Eq. (4) and (6) we obtain:

$$\sum_{i=j \ldots l-1} 2^i x^-_i \ < \ \sum_{i=j+1 \ldots l-1} 2^i c_i + 2^j \tag{7}$$

By Eq. (3) and (5) we obtain:

$$2^l + \sum_{i=j+1 \ldots l-1} 2^i c_i \ < \ \sum_{i=j \ldots l-1} 2^i x^+_i \tag{8}$$

Eq. (7) and (8) give:

$$2^l - 2^j \ < \ \sum_{i=j \ldots l-1} 2^i (x^+_i - x^-_i) \tag{9}$$

Therefore, bounding the right-hand side:

$$2^l - 2^j \ < \ \sum_{i=j \ldots l-1} 2^i \tag{10}$$

But:

$$\sum_{i=j \ldots l-1} 2^i = \sum_{i=0 \ldots l-1} 2^i - \sum_{i=0 \ldots j-1} 2^i = (2^l - 1) - (2^j - 1)$$

which contradicts Eq. (10).

3.2 The upper bound of x_l has a support

The proof is completely symmetric to the one for the lower bound of x_l.

3.3 Both bounds of every $x_i,\ i \neq l$, have a support

We know that there exists a solution that assigns value 0 to x_l and a solution that assigns value 1 to x_l. In other words there exist two tuples t_1 and t_2 of the form:

$$t_1 = \langle c_n, \ldots, c_{l+1}, 0, a_{l-1} \ldots a_0 \rangle$$
$$t_4 = \langle c_n, \ldots, c_{l+1}, 1, b_{l-1} \ldots b_0 \rangle$$

such that $x^- \leq \mathit{eval}(t_1) \leq \mathit{eval}(t_4) \leq x^+$. These tuples provide a support for the value $c_i = x_i^- = x_i^+$ of each variable $x_i,\ i > l$. Now the tuples:

$$t_2 = \langle c_n, \ldots, c_{l+1}, 0, x_{l-1}^+ \ldots x_0^+ \rangle$$
$$t_3 = \langle c_n, \ldots, c_{l+1}, 1, x_{l-1}^- \ldots x_0^- \rangle$$

are such that $x^- \leq \mathit{eval}(t_1) \leq \mathit{eval}(t_2) \leq \mathit{eval}(t_3) \leq \mathit{eval}(t_4) \leq x^+$. We have exhibited a support (t_3) for the lower bounds of every variable $x_i,\ i < l$ and a support (t_2) for the upper bounds of these variables.

4. X is not reduced optimally

Our second result states that in general propagation does *not* reduce the bounds of x in an optimal way. More precisely, we prove that the intervals computed by bound propagation can be *twice as large as they should ideally.*

Proposition 2 *There exists an infinite family of instances for which the bounds of X are not consistent after bound propagation; moreover the width of the interval of values for X after propagation can be arbitrarily close to twice the width of the optimally reduced interval.*

This result shows that some improvement is possible. We start by exhibiting an example where the over-approximation of the bounds of X happens:

Example 1 *We consider an 8-bit version of the constraint:*

$$X = 128x_7 + 64x_6 + 32x_5 + 16x_4 + 8x_3 + 4x_2 + 2x_1 + 1x_0$$

Now let the ranges be defined as follows:

$$X \in [64, 191],\ \ x_7, x_6 \in [0, 1], x_5, x_4, \ldots x_0 \in [1, 1]$$

Note that the binary representation of 64 is $\langle 01000000 \rangle$ and the representation of 191 is $\langle 10111111 \rangle$.

The previous ranges are stable under propagation. For instance $64 \geq 0 + 0 + 32 + 16 + 8 + 4 + 2 + 1$, and all the other inequalities of Eq. 2 are also satisfied. Yet value $X = 64$ is not consistent, since the only assignment of the x_is that evaluates to 64 needs the values $x_i = 0$ for $i \leq 5$. Indeed, the smallest consistent value larger than 64 is $\langle 01111111 \rangle$, i.e., 127.

The example can be generalized: if we take:

- $X^- = 2^{n-1}$ (*i.e.*, $X^- = \langle 01000000\ldots\rangle$);
- $X^+ = 2^{n+1} - 1 - 2^{n-1}$ (*i.e.*, $X^+ = \langle 101111111\ldots\rangle$);
- $x_n^- = 0$; $x_n^+ = 1$; $x_{n-1}^- = 0$; $x_{n-1}^+ = 1$; (*i.e.*,the two highest-weight bits are not instantiated);
- $x_i^- = x_i^+ = 1$, for $i \in 1..n-2$ (*i.e.*, the lower-weight bits are fixed to value 1).

Then we have bounds that are stable under propagation, but X^- nevertheless has no support. The lowest value for X that is consistent is obtained by switching all the rightmost bits of X^- to 1, giving the value $2^n - 1$. The width of the ideal, bound-consistent interval is $2^{n+1} - 1 - 2^{n-1} - (2^n - 1) = 2^{n-1}$ The width of the intervals stable under propagation is: $2^{n+1} - 1 - 2^{n-1} - 2^{n-1} = 2^n - 1$. We have therefore exhibited, for each size n, an instance where the over-approximation is:

$$\frac{2^n - 1}{2^{n-1}}$$

which is getting infinitely close to 2 as n increases.

5. An improved propagator

We now show how the bounds of X can be reduced optimally using a simple linear-time procedure. This additional step can be performed after the bounds of the x_is have been reduced by means of classical bound propagation, and we therefore have an optimal reduction of all intervals.

The algorithm works as follows: let $\langle l_n \ldots l_0\rangle$ be the bits of X^- and $\langle r_n \ldots r_0\rangle$ be the bits of X^+, *i.e.*,

$$\begin{array}{rcl} X^- & = & \sum_{i\in 0\ldots n} 2^i l_i \\ X^+ & = & \sum_{i\in 0\ldots n} 2^i r_i \end{array}$$

We shall simply correct X^- and X^+ so that their bits all take values that fall within the domains of the x_is. To do that we compute new vectors of values $\langle l'_n \ldots l'_0\rangle$ and $\langle r'_n \ldots r'_0\rangle$. Each l'_i and r'_i is defined as follows, for $i \in 0\ldots n$:

$$l'_i = \begin{cases} 1 & \textit{if } x_i^- = 1 \\ l_i & \textit{otherwise} \end{cases} \qquad r'_i = \begin{cases} 0 & \textit{if } x_i^+ = 0 \\ r_i & \textit{otherwise} \end{cases}$$

We last assign X^- to $\sum_{i\in 0\ldots n} 2^i l'_i$ and X^+ to $\sum_{i\in 0\ldots n} 2^i r'_i$. It is clear that we have lost no solution in the reduction, since the bits of the l_is and r_is that we modified were not set correctly. It is also easy to see that the bounds are now consistent: the value of the support of X^- (*resp.* X^+) for variable x_i is directly given by l'_i (*resp.* r'_i).

References

J. G. Cleary. Logical arithmetic. *Future Computing Systems*, 2(2):125–149, 1987.

E. Davis. Constraint propagation with interval labels. *Artificial Intelligence*, 32(3):281–331, 1987.

W. Harvey and P. J. Stuckey. Improving linear constraint propagation by changing constraint representation. *Constraints*, 8(2):173–207, 2003.

M. A. Trick. A dynamic programming approach for consistency and propagation for knapsack constraints. In *Proc. of Int. Conf. on Integration of AI and OR Techniques in CP for Combinatorial Optimisation Problems (CP-AI-OR)*, 2001.

Z. Yuanlin and R. H. C. Yap. Arc consistency on n-ary monotonic and linear constraints. In *Proc. of Int. Conf. on Principles and Practice of Constraint Programming (CP)*, pages 470–483. Springer, 2000.