Research on Mechanism of Variable Event in CLP System

Jing Wan, Likang Tian, Junkai Yi and Yubin Zhou
Department of Information Science and Technology, Beijing University of Chemical Technology, Beijing 100029, China

Abstract: Constraint inference engine as the core part of constraint logic programming system comprises variable set, temporary container and constraint filter and inference engine and adopts branching strategy, exploration strategy and node backtracking strategy to complete the inference task. This study introduces variable event to reduce the triggering times of constraint filter and analyzes the trigger probability of inference engine. The experiments show that variable event setting can enhance the efficiency of constraint inference engine.

Keywords: Constraint logic programming, inference engine, variable event

INTRODUCTION

Constraint logic programming, which combines the advantages of consistency algorithms and heuristic search algorithms in artificial intelligence, is one of the important paradigms for solving discrete domain constraint problems (Hendler et al., 2006).

Generally, the basic strategy adopted by constraint logic programming is binary tree search, which begins from root node in a prescribed way. Once a leaf is found, an answer is found. The establishment of binary tree is a modeling process of practical problem, which involves a series of logic variables as the mathematical foundation for problem description. Each variable has its own scope referred to as finite field. Constraint conditions that should be met by problem description can be expressed as inter-variable logic constraint, which is a constraint filter in the search process. What needs to be solved next is how to get a smaller finite field so as to satisfy constraint filter. And then the inference engine of constraint logic programming is initiated. Inference engine can perform binary tree search according to the prescribed strategy in a temporary container. Interactions of variable set, constraint filter, temporary container and inference engine collectively constitute constraint inference engine (Miguel et al., 2008).

Efficiency improvement of constraint inference engine has become one of the main research directions for constraint logic programming (Roman et al., 2010). In order to improve the search efficiency of constraint logic programming system, enhancement of the pruning ability of constraint inference engine is the main approach (Alan et al., 2009), which involves such strategies as branch algorithm (Asma et al., 2011), exploration orientation (Samir et al., 2010) and node backtracking algorithm (Mesyagutov et al., 2012).

Based on these strategies, this study proposes the classification of variable compression event, which reduces the accelerated cycles of trigger times of constraint filter in the inference engine and improves search efficiency.

VARIABLE EVENT

Search efficiency analysis of constraint inference engine: Constraint inference engine is composed of variable set, constraint filter, temporary container and inference engine. Constraint inference engine has branching strategy, exploration strategy and node backtracking strategy, which prescribes the behaviors of binary tree search. In constraint logic programming, constraint inference engine is the core, whose efficiency improvement is important to enhance the search efficiency. This study proposes the classification of variable events in the purpose of improving search efficiency by reducing the search time.

Each inference of constraint inference engine does not require the involvement of all constraint filters. Only when a variable’s finite field is changed will the constraint filter related to this variable be triggered and participate in the inference. This is the trigger mechanism of filter for all constraint logic programming systems. A reduction in finite field of a variable is defined as a variable event, denoted as Ex. In order to further enhance the search efficiency, variable event needs to be processed, by classifying the constraint filters with respect to a variable according to the type of variable event. Only when a specific variable event occurs will the related constraint filter be
triggered. Variable event setting does not increase memory consumption but avoids many unnecessary trigger of constraint filter.

**Type of variable event**: Suppose the initial discrete range of \( x \) is \( \{w_1, w_2, \ldots, w_n\} \), then the range of \( x \) will be reduced after \( w_k \) is filtered out by constraint inference engine. Variable \( x \) before and after the filtering of constraint inference engine can be expressed by Formula (1) below:

\[
x \in \{w_1, w_2, \ldots, w_n\} \rightarrow x \in \{w_1, \ldots, w_k, w_{k+1}, \ldots, w_n\}
\]

As we can see from Formula (1), when some values are arbitrarily filtered, the form of the finite field of a variable is changed. Then event type \( E_0 \) of variable \( x \) is the variation of finite field; \( E_0 \) is defined as DOMCHG by definition 1.

**Definition 1**: For \( x \) and its finite field \( D_0 \), if \( D_0' \subset D_0 \) and \( D_0' \neq \emptyset \), then \( E_0 \) is defined as DOMCHG in the process of \( D_0 \rightarrow D_0' \).

As shown by Formula (2) and (3), if value \( w_i \) or \( w_k \) is filtered from variable \( x \), constraint inference engine will define the type of variable event \( E_0 \) as BOUND, with a changed boundary of variable range:

\[
x \in \{w_1, w_2, \ldots, w_n\} \rightarrow x \in \{w_1, \ldots, w_{k-1}, w_{k+1}, \ldots, w_n\} \tag{2}
\]

\[
x \in \{w_1, w_2, \ldots, w_n\} \rightarrow x \in \{w_1, \ldots, w_{k-1}w_{k+1}, \ldots, w_n\} \tag{3}
\]

For finite field \( D \), suppose \( \max(D) \) and \( \min(D) \) are respectively the maximum value and the minimum value and then there is definition 2.

**Definition 2**: For \( x \) and its finite field \( D_0 \), if \( D_0' \subset D_0 \), \( D_0' \neq \emptyset \) and either of the following two conditions are satisfied: \( \max(D_0') < \max(D_0) \) or \( \min(D_0') > \min(D_0) \), then \( E_0 \) is defined as BOUND in the process of \( D_0 \rightarrow D_0' \).

If the range of variable \( x \) is reduced to only one value from many discrete values, constraint inference engine will define the type of variable event \( E_0 \) as SINGLE, which is expressed by Formula (4):

\[
x \in \{w_1, w_2, \ldots, w_n\} \rightarrow x \in \{w_k\} \quad (1 \leq k \leq n) \tag{4}
\]

For finite field \( D \), suppose \( \text{Size}(D) \) the represents number of possible values in \( D \), then there is definition 3.

**Definition 3**: For \( x \) and its finite field \( D_0 \), if \( D_0' \subset D_0 \), \( \text{Size}(D_0') = 1 \), then \( E_0 \) is defined as SINGLE in the process of \( D_0 \rightarrow D_0' \).

Therefore, variable event \( E \) has three types: SINGLE, BOUND and DOMCHG. From the above three definitions, the following two inferences can be obtained:

**Inference 1**: For \( x \) and its finite field \( D_0 \), if \( D_0' \subset D_0 \) and \( D_0' \neq \emptyset \) and \( E_0 \) is defined as SINGLE in the process of \( D_0 \rightarrow D_0' \), then \( E_0 \) can also be defined as DOMCHG and BOUND simultaneously.

**Inference 2**: For \( x \) and its finite field \( D_0 \), if \( D_0' \subset D_0 \), \( D_0' \neq \emptyset \) and \( E_0 \) is defined as BOUND in the process of \( D_0 \rightarrow D_0' \), then \( E_0 \) can be defined as DOMCHG.

Here we can see that \( E_0 \) can be defined as more than one variable event simultaneously, but that is not to say that \( E_0 \) is changeable. What is true is that a variable defined as a certain event type can contain other events simultaneously. For variable \( x \), \( \text{SINGLE} \subseteq \text{BOUND} \subseteq \text{DOMCHG} \), \( \text{SINGLE} \) is contained in \( \text{BOUND} \) and \( \text{DOMCHG} \), while \( \text{BOUND} \) is contained in \( \text{DOMCHG} \).

**VARIABLE EVENT SETTING**

When constraint filters are incorporated into constraint inference engine, relevant variables need to be standardized and initialized so that the variables can meet the requirements of constraint inference engine. In the initialization step, variable event setting can be performed according to actual contents of constraint algorithm.

From the above analysis we have known that variable event can have three types: SINGLE, BOUND and DOMCHG. In this section, variable event setting for a specific constraint filter is discussed by taking linear filter as an example. Formula (5) represents a linear constraint filter; \( a_i \) is a non-zero integer constant:

\[
\sum_{i=0}^{i<n} a_i x_i = A \tag{5}
\]

where, \( A \) is an integer constant. The left side of Formula (5) can be divided into two parts according to the sign of \( a_i \), as shown in Formula (6):

\[
A = \sum_{i=0}^{i<n} a_i x_i = \sum_{a_i>0} a_i x_i + \sum_{a_i<0} a_i x_i \tag{6}
\]

Suppose \( \max(x) \) and \( \min(x) \) respectively represents the maximum value and minimum value of \( x \). Then from Formula (6), we can see that the range of \( x_i \) is:
### Trigger times analysis of variable event:

The trigger probability of variable event is mainly reflected in the trigger times of constraint filter. Suppose the finite field of variable \( x \) is \( D_x = \{w_1, w_2, \ldots, w_k\} \), where \( w_1 < w_2 < \ldots < w_k \) and they are all integer numbers. Then, when DOMCHG happens to \( x \), possible compression of \( D_x \) will occur as follows:

\[
C_1^k + C_2^k + \ldots + C_k^{k-1} = 2^k - 2
\]

At this moment, the probability of \( x \) triggering related filter is \( O(2^k) \). When BOUND happens to \( x \), possible compression of \( D_x \) will occur as follows:

\[
k + (k - 1) + \ldots + 1 - 1 = \frac{k(k + 1)}{2} - 1
\]

At this moment, the probability of \( x \) triggering related filter is \( (k^2) \). When SINGLE happens to \( x \), there are \( k \) types of possible compression of \( D_x \); \( O(k) \) is the probability.

If DOMCHG is a total probability event, which can be defined as:

\[
P(DOMCHG) = 1
\]

Then occurrence probability of SINGLE and BOUND is:

\[
P(BOUND | DOMCHG) = \frac{k(k + 1)}{2} - 1
\]

\[
P(SINGLE | DOMCHG) = \frac{k^2 + k - 2}{2^{k+2}} - 4
\]

As we can see from the above formula, for a certain variable, the greater the value of \( k \), the less the occurrence of SINGLE than that of DOMCHG and BOUND and the less the occurrence of BOUND is than that of DOMCHG.

<table>
<thead>
<tr>
<th>Sample</th>
<th>Nqueen (100)</th>
<th>Magic square (5)</th>
<th>Hamilton</th>
<th>Photo</th>
<th>Robin</th>
<th>Larry</th>
<th>Bridge</th>
</tr>
</thead>
<tbody>
<tr>
<td>Trigger times before the introduction</td>
<td>102</td>
<td>87</td>
<td>233</td>
<td>302</td>
<td>156</td>
<td>126</td>
<td>119</td>
</tr>
<tr>
<td>Trigger times after the introduction</td>
<td>98</td>
<td>76</td>
<td>201</td>
<td>237</td>
<td>69</td>
<td>112</td>
<td>67</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Sample</th>
<th>Nqueen (100)</th>
<th>Magic square (5)</th>
<th>Hamilton</th>
<th>Photo</th>
<th>Robin</th>
<th>Larry</th>
<th>Bridge</th>
</tr>
</thead>
<tbody>
<tr>
<td>T before the introduction</td>
<td>0.53</td>
<td>0.15</td>
<td>22.13</td>
<td>26.05</td>
<td>12.47</td>
<td>3.98</td>
<td>1.14</td>
</tr>
<tr>
<td>T after the introduction</td>
<td>0.45</td>
<td>0.14</td>
<td>15.68</td>
<td>18.46</td>
<td>5.75</td>
<td>2.54</td>
<td>0.67</td>
</tr>
</tbody>
</table>

Table 1: Comparison of trigger times before and after the introduction of variable event

Table 2: Comparison of operation time before and after the introduction of variable event (operation time: s)
According to the previous method for constraint filter trigger, the probability of \( x \) triggering related SINGLE or BOUND filter is \( O(2^k) \). The probability analysis of variable event occurrence shows that: when DOMCHG happens to \( x \), BOUND and SINGLE constraint filter will not be triggered; when BOUND happens to \( x \), SINGLE constraint filter will not be triggered.

EXPERIMENTAL RESULT ANALYSIS

After the introduction of variable event, the constraint filters are much less triggered during the search process. The experimental environment for constraint logic programming is as follows: HP2210b, T7300, duo CPU 2.0 G, 1 G Memory.

For the sake of comparison, classical problems are used for constraint solving. The experimental results are shown in Table 1 and 2. Operation time in the experiment includes the display time of screen and therefore the actual solution time is probably less. The introduction of variable event is of great help for improving the search efficiency of constraint logic programming and reducing unnecessary trigger of constraint filter.

CONCLUSION

Constraint inference engine as the core part of constraint logic programming comprises variable set, temporary container and constraint filter and inference engine and adopts branching strategy, exploration strategy and node backtracking strategy to complete the inference task. Search time is one of the main performance indexes of constraint logic programming system. This study introduces variable event to reduce the triggering of constraint filter and analyzes the trigger probability of inference engine after the introduction of variable event. Variable event can be divided into three types: SINGLE, BOUND and DOMCHG. Accordingly, constraint filter can be also divided into three types depending on the type of variable event that triggers the constraint filter: SINGLE, BOUND and DOMCHG. We have experimentally verified that variable event setting can enhance the efficiency of constraint inference engine.

REFERENCES


