УДК 004.82

Сравнение способов представления неточных значений в методе недоопределённых вычислений

Сидоров В.А. (Институт систем информатики СО РАН)

Метод недоопределённых вычислений является одним из подходов для решения задачи удовлетворения ограничений. Его базовым понятием является понятие вида недоопределённости, которое представляет собой способ задания неточного значения в виде множества. В статье рассматриваются различные виды недоопределённости, приводятся их формальные определения, выполняется сравнение и выделяются их основные свойства.

***Ключевые слова**: задача удовлетворения ограничений, недоопределённые вычисления, интерввальное значение, мультиинтервал.*

1. Введение

Задача удовлетворения ограничений (**ЗУО**) является популярным подходом для решения нестандартных и сложных вычислительных задач. ЗУО базируется на понятиях «*ограничение*» и «*переменная*», с помощью которых можно описать широкий класс задач. Впервые задача удовлетворения ограничений была сформулирована в начале 70х годов [11, 14]. К настоящему времени разработано множество методов решения ЗУО и с их помощью реализованы как отдельные приложения, так и целые системы программирования, успешно применяемые на практике.

В начале 80х годов А.С. Нариньяни был предложен *метод недоопределённых вычислений* [2]. Метод недоопределённых вычислений является одним из методов решения задачи удовлетворения ограничений и может быть описан следующим образом:

- Модель (описание задачи) состоит из набора переменных и ограничений.
- С каждой переменной связывается множество допустимых значений (*недоопределённое значение*).
- Ограничения связывают переменные и позволяют вычислять новые (недоопределённые) значения для своих аргументов.
- Алгоритм вычислений имеет потоковый характер, выражающийся в том, что изменение значения переменной активирует (вызывает к исполнению) ограничения,

связанные с данной переменной. В свою очередь, исполнение (удовлетворение) ограничения может вызывать изменение связанных с ним переменных.

Важнейшей особенностью метода недоопределённых вычислений является его универсальность — метод позволяет решать задачи, содержащие как непрерывные, так и дискретные значения переменных. Более того, тип переменных может быть полностью произвольным; для работы метода достаточно определить операцию пересечения для выбранного представления множества значений этой переменной. Таким образом переменная описывается не только её типом данных, но и способом представления множества её значений, который мы будем называть *видом недоопределённости*.

От выбора вида недоопределённости сильно зависит как скорость вычисления отдельных ограничений, так и точность представления множества значений. Однако сравнение видов недоопределённости обычно ограничивается утверждениями «интервальное представление грубее перечисления», «перечисление слабее мультиинтервального представления» и т.д. (например, в статье [6]) - которые часто некорректны.

В данной работе приводятся формальные определения основных видов недоопределённости, используемых при решении задачи удовлетворения ограничений и проводится их сравнительный анализ.

1. Задача удовлетворения ограничений и метод недоопределённых вычислений

Опишем область применения видов недоопределённости: приведём краткое описание метода недоопределённых вычислений и задачи удовлетворения ограничений.

1.1 Задача удовлетворения ограничений

Задача удовлетворения ограничений может быть сформулирована с различной степенью детализации [3, 9, 10]. Далее в статье будем использовать следующие определения:

Задача удовлетворения ограничений M – это тройка $\langle Var, D, C \rangle$, где:

- $Var = \{v_1, \ldots, v_n\}$ – множество переменных.
- $D = \langle D_{v1}, \ldots, D_{vn} \rangle$ – набор множеств значений переменных.
- $C = \{c_1, \ldots, c_k\}$ – множество ограничений. Ограничение $c\left(R_c, \langle v_{c1}, \ldots, v_{cm} \rangle\right)$ задаётся некоторым отношением $R_c(D_1, \ldots, D_m) \subseteq D_1 \times \ldots \times D_m;\ D_i \subseteq D_{vi}$, которое определено

над множествами значений переменных $v_{c1},\ldots,v_{cm}$ и является подмножеством картезианского произведения этих множеств.

Означивание переменной – это функция присваивания (множественного) значения переменной: $val(v,d):Var\times D\to D_v;\ D_v\in D$.

Набор множеств значений $\langle A_1,\ldots,A_m\rangle, A_i\subseteq D_i$ ***совместен*** на ограничении $c=\left(R_c,\langle v_{c1},\ldots,v_{cm}\rangle\right)$, если $\exists d_1\in A_1,\ldots,\exists d_m\in A_m$, такие что $\langle val(v_{c1},d_1),\ldots,val(v_{cm},d_m)\rangle\in R_c$.

Локально-совместное решение задачи удовлетворения ограничений – множество значений всех переменных, совместное на каждом ограничении модели.

Решение (глобальное) задачи удовлетворения ограничений — множество значений всех переменных, совместное на всех ограничениях модели одновременно.

Точное решение задачи удовлетворения ограничений — это глобальное решение $\langle A_1,\ldots,A_n\rangle$, состоящее из одно-элементных множеств: $A_i=\{x_i\};x_i\in D_{vi}$.

Отметим, что большинство алгоритмов решения ЗУО ищет локально-совместное решение, представленное множеством значений переменной, в то время как пользователю обычно нужно точное решение.

1.2 Метод недоопределённых вычислений

Для описания метода недоопределённых вычислений уточним ЗУО для множественного представления значений переменных.

Недоопределённое значение D_v^* переменной v является подмножеством множества значений D_v.

Экстенсиональное задание ограничения (в виде множества всех допустимых комбинаций значений аргументов), приведённое в определении ЗУО, заменяется на интенсиональное. Для этого для ограничения $c\left(R_c,\langle v_{c1},\ldots,v_{cm}\rangle\right)$ определяются ***функции интерпретации*** $f_{c,i}:D_{v1}^*\times\ldots,\times D_{vm}^*\to D_{vi}^*$, которые вычисляют новое значения каждого i-ого аргумента ограничения $x'_i:=f_{c,i}\left(x_1,\ldots,x_m\right)$, и удовлетворяют следующим свойствам:

- $f_{c,i}$ сужает множество значений x_i: $x'_i\subseteq x_i$.
- x'_i - максимальное совместное подмножество x_i. То есть, если $\exists y\subseteq D_{vi}^*$ такое, что $\langle x_1,\ldots,x_{i-1},y,x_{x+1},\ldots,x_m\rangle\subseteq R_c$ и $y\subseteq x_i$, то $y\subseteq x'_i$.

Заметим, что способ представления множества (а все аргументы и результат функции интерпретации являются множествами) сильно влияют как на реализацию самой функции, так и на её результат.

Для описания алгоритма работы метода недоопределённых вычислений введём дополнительные обозначения:

- $d(t) \subseteq \langle D_{v1}^*, \ldots, D_{vn}^* \rangle$ - множество значений всех переменных на шаге t.
- $d(t,i) \subseteq D_{vi}^*$ - множество значений i-ой переменной на шаге t.
- $Arg(c) = \{v_{c1}, \ldots, v_{cm}\}$ - множество переменных, являющихся аргументами ограничения c.
- $d(t)|_c = \langle d(t,c1), \ldots, d(t,cm) \rangle$ - множество значений всех переменных, являющихся аргументами ограничения $c(R_c, \langle v_{c1}, \ldots, v_{cm} \rangle)$ на шаге t.

Вход

Множество ограничений $C = \{c_1, \ldots, c_k\}$, заданных с помощью функций интерпретаций; множество переменных $Var = \{v_1, \ldots, v_n\}$; начальные значения переменных $\langle D^*_{v1}, \ldots, D^*_{vn} \rangle$.

Выход

Новые значения переменных $\{v_1, \ldots, v_n\}$.

Алгоритм

$t := 0$

// Инициализация начальных значений переменных d(t) и очереди ограничений Q(t)

$d(t) := \langle D^*_{v1}, \ldots, D^*_{vn} \rangle$; $Q(t) := C$

// Проверка условия останова:

// 1. Очередь ограничений не пуста

// 2. Значения всех переменных не пусты

DO WHILE $Q(t) \neq \varnothing \wedge \forall i : d(t,i) \neq \varnothing$

// Выбор ограничения из очереди

$c \in Q(t)$

// Удовлетворение ограничения

$$d(t+1,i) := \begin{cases} d(t,i)\ if v_i \notin Arg(c) \\ f_{c,j}(d(t)|_c)\ if v_i \in Arg(c) \wedge i = cj \end{cases}$$

// Выделение изменившихся переменных

$\{v'_i\} : v'_i \in Var;\ d(t+1,i) \neq d(t,i)$

// Выделение связанных с ними ограничений

$\{c'_j\} : c'_j \in C;\ Arg(c'_j) \cap \{v'_i\} \neq \varnothing$

// Добавление новых ограничений в очередь

$Q(t+1) := Q(t) \cup \{c'_j\} \setminus \{c\}$

// Следующий шаг

$t := t + 1$

END DO

Таблица 1. Алгоритм решения ЗУО методом недоопределённых вычислений.

Если в результате работы алгоритма множество значений любой из переменных стало пусто, то система несовместна. Иначе, множество текущих значений переменных является локально-совместным.

2. Виды недоопределённости

Так как метод недоопределённых вычислений работает с множествами значений, то важнейшим вопросом является способ представления множества, называемый *видом недоопределённости*.

Вид недоопределённости SD переменной v это множество подмножеств значений из области допустимых значений D_v, удовлетворяющее следующим свойствам:

1. $SD(D_v) \subseteq 2^{D_v}$.
2. $\varnothing \in SD(D_v),\ D_v \in SD(D_v),\ \forall x \in D_v, \{x\} \in SD(D_v)$.
3. $x_1, x_2 \in SD(D_v) \Rightarrow x_1 \cap x_2 \in SD(D_v)$.

Свойство 1 говорит, что вид недоопределённости является способом представления множества.

Свойство 2 задаёт необходимый набор подмножеств:

- пустое множество (для индикации несовместности текущего значения переменной с одним из ограничений);
- всё множество допустимых значений переменной (для задания начального значения и области определения переменной);
- все точные значения (для возможности нахождения любого точного решения из области определения переменной).

Свойство 3 ограничивает набор используемых алгоритмов решения ЗУО такими, которые монотонно сужают множество допустимых значений переменных.

Для видов недоопределённости естественным образом задаётся частичный порядок:

Вид недоопределённости $SD_1(D)$ ***более полный***, чем $SD_2(D)$ ($SD_2(D) \leq_P SD_1(D)$), если $SD_2(D) \subseteq SD_1(D)$.

Виды недоопределённости $SD_1(D)$ и $SD_2(D)$ ***эквивалентны*** ($SD_2(D) =_P SD_1(D)$), если $SD_2(D) = SD_1(D)$.

Вид недоопределённости $SD(D)$ ***полный***, если $\forall X \subseteq D \Rightarrow X \in SD(D)$.

Свойство полноты влияет на эффективность вычислений:

- Нахождение точного решения задачи удовлетворения ограничений является NP-сложной задачей.
- Для нахождения локально-совместного решения разработан ряд эффективных алгоритмов с полиномиальной сложностью [13].
- Соответственно, более полный вид недоопределённости позволяет более точно представлять локально-совместное решение и, как следствие, выполнять более медленный поиск точного решения ЗУО на более узком множестве значений.

Рассмотрим некоторые виды недоопределённости, реализованные в системах Nemo+ и NemoNext [1, 4].

3.1 Одиночное значение

Вид недоопределённости $Single$:

$Single(D) = \{\varnothing, D, \{x\} \mid x \in D\}$.

Single является минимальным видом недоопределённости: любое дальнейшее уменьшение количества элементов множества приведёт к нарушению свойства 2 определения вида недоопределённости. Соответственно, *Single* является подмножеством любого другого вида недоопределённости.

На практике, *Single* часто используется в следующих случаях:

- Работа с узкими областями значений: $|D| \approx 2$.
- Представление точных констант.
- Моделирование специфических сетевых задач, таких как вычислительные сети Тыугу [7].

3.2 Интервал

В системах, решающих задачу удовлетворения ограничений для непрерывных областей [3, 8, 12], обычно используется интервальное представление множества значений.

Пусть на D задано отношение частичного порядка. Обозначим $[low, upp]_D = \{x \mid low, upp, x \in D; low \le x \le upp\}$. Тогда вид недоопределённости *Interval* определяется следующим образом:

$$Interval(D) = \{\varnothing, [low, upp]_D \mid \forall low, upp \in D; low \le upp\} =$$
$$= \{\varnothing, \{X \mid X \subseteq D; \exists low, upp \in X : \forall x \in X \quad low \le x \le upp\}\}.$$

Отметим, что представление в виде интервала возможно, только если D является решёткой: для каждого подмножества $X \subseteq D$ существует точная верхняя и нижняя грани:

$$\forall X \subseteq D; \exists \sup(X) \in D, \exists \inf(X) \in D : \forall x \in X \ \inf(X) \le x \le \sup(X).$$

Это свойство необходимо для замкнутости *Interval* относительно операции пересечения:

Утверждение. *Interval* замкнут относительно операции пересечения.

Пусть $x, y \in Interval(D)$, $x = \lfloor x_{low}, x_{upp} \rfloor$, $y = [y_{low}, yupp]$. Тогда:

$$x \cap y = \{a \mid a \in D, x_{low} \le a \le x_{upp}\} \cap \{b \mid b \in D, y_{low} \le b \le y_{upp}\} = \{c \mid c \in x, c \in y\} =$$

$$= \{c \mid c \in D, x_{low} \le c \le x_{upp}, y_{low} \le c \le y_{upp}\} =$$

$$= \{c \mid c \in D, \inf(x_{low}, y_{low}) \le c \le \sup(x_{upp}, y_{upp})\} \in Interval(D).$$

Последнее равенство вытекает из определения точной верхней и нижней грани; $\inf(x_{low}, y_{low}) \in D$, $\sup(x_{upp}, y_{upp}) \in D$, т. к. D – решётка.

3.3 Перечисление

В системах, решающих задачу удовлетворения ограничений для дискретных областей [15], допустимое значение обычно представляют в виде прямого перечисления значений.

Вид недоопределённости *Enum*:

$$Enum(D) = \{\varnothing, X \mid X \subseteq D\}.$$

Вид недоопределённости *Enum* является полным (так как содержит все подмножества D).

Теоретически, *Enum* может применяться к любым, не только к счётным D. Но на практике, *Enum* не применим как для непрерывных, так и для больших дискретных множеств значений из-за больших затрат на хранение и обработку значений. Поэтому при реализации перечисления размер множества принудительно ограничивают, получая другой вид недоопределённости - *ограниченное перечисление*.

3.4 Ограниченное перечисление

Если количество элементов *Enum* ограничиваются некоторым числом N, то получим следующий вид недоопределённости:

Вид недоопределённости $Enum_N$:

$$Enum_N(D) = \{\varnothing, D, X \mid X \subseteq D, |X| < N\}$$

Заметим, что $Enum_N$ совпадает с $Enum$ при $N \geq |D|$. В противном случае, $Enum_N$ не является полным, так как любое подмножество из более чем N элементов (кроме всего D) не представимо в $Enum_N$.

3.5 Мультиинтервал

Представление значения в виде мультиинтервала было введено в работах [5, 6] для того, чтобы совместить преимущества *Interval* (компактность, эффективность, работа с непрерывными областями) и *Enum* (полнота представления).

Вид недоопределённости *Multiinterval*:

$$MultiInterval(D) = \{X \mid X = \cup X_i,\ X_i \in Interval(D)\}.$$

Вид недоопределённости *Multiinterval* является полным (так как $Enum \subseteq Multiinterval$)

Несмотря на то, что мультиинтервал является более компактным представлением, чем $Enum$, на практике также приходится ограничивать количество составляющих его подинтервалов.

3.6 Ограниченный мультиинтервал

Множественное представление $Multiinterval_N$:

$$MultiInterval_N(D) = \{X \mid X = \cup X_i,\ X_i \in Interval(D),\ i \leq N\}.$$

Такое представление не является видом недоопределённости, так как не замкнуто относительно операции пересечения при $N > 1$: $x, y \in MultiInterval_N(D) \Rightarrow x \cap y \in MultiInterval_{2*N-1}(D)$. Этот факт является причиной принципиальных проблем мультиинтервального представления:

- Автоматическое увеличение количества подинтервалов в процессе решения ЗУО.
- Неоднозначность представления множества значений с помощью $Multiinterval_N$.

3.7 Смешанный вид недоопределённости

Альтернативный способ совместить преимущества интервального представления и перечисления представляет собой комбинацию двух значений, одно из которых является $Interval(D)$, другое – $Enum_N(D)$:

Вид недоопределённости $Mixed_N$:

$$Mixed_N(D) = \{I, E \mid I \in Interval(D),\ E \in Enum_N(D)\} = \{X \mid X \subseteq I \cap E\}.$$

3.8 Вид недоопределённости для составных значений

Отдельно следует рассмотреть случай, когда значение переменной содержит внутри себя отдельные поля (является составным): $D = D_1 \times \ldots \times D_n = \{d_1, \ldots, d_n \mid d_1 \in D_1, \ldots, d_1 \in D_n\}$.

Для таких значений можно определить специальный вид недоопределённости *Struct* :

$Struct(D) = SD_1(D_1) \times \ldots \times SD_n(D_n) = \{x_1, \ldots, x_n \mid x_i \in SD_i(D_i)\}$.

То есть, недоопределённое составное значение является комбинацией недоопределённых значений его элементов.

3. Свойства видов недоопределённости

Рассмотрим некоторые свойства видов недоопределённости.

4.1 Мощность видов недоопределённости

Если D - конечное множество, то для каждого вида недоопределённости можно явно выписать его мощность:

- $|Single(D)| = |D| + 2$.
- $|Interval(D)| = 1 + |D| * (|D| + 1)/2$.
- $|Enum(D)| = 2^{|D|}$.
- $|Enum_N(D)| = 2 + \sum_{i=1}^{i=N} C_{|D|}^i$.
- $|Mixed_N(D)| = 2 + \sum_{i=1}^{i=N} C_{|D|}^i + |D| * (|D| - N)/2$.
- $|Multiinterval(D)| = 2^{|D|}$.

Заметим, что при $|D| = 2$ (например, $D = \{false, true\}$) мощности всех видов недоопределённости совпадают.

4.2 Полнота видов недоопределённости

В общем случае мощность вида недоопределённости оценить невозможно, но можно сравнить виды недоопределённости по полноте. Построим иерархию видов недоопределённости по этому критерию.

Раннее мы уже отметили некоторые свойства видов недоопределённости:

- Вид недоопределённости *Single* является минимальным по полноте.
- Вид недоопределённости *Enum* является полным.

- Вид недоопределённости *Multiinterval* также является полным.
- Виды недоопределённости $Enum_N$ и *Interval* не сравнимы по полноте:

Пусть $D_x \subset ; x = \{1,2,4,5,6\}$. Тогда $Interval\ (x) = [1,6\ ]$; $Enum_4(x) =$; $Enum_5(x) = x$. И, соответственно, $Enum_5(x) \subset Interval(x) \subset Enum_4(x)$.

Для различных пар видов недоопределённости можно выписать соотношения:

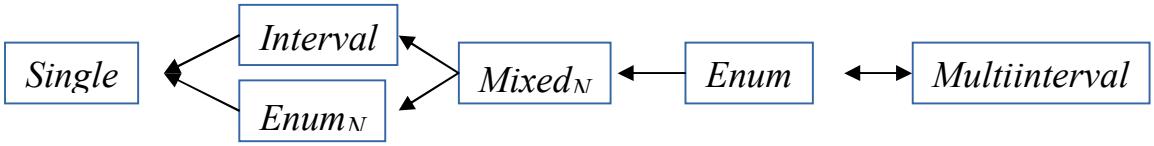
- $Single(D) \le_P Interval(D)$.
- $Interval(D) \le_P Enum(D)$.
- $Enum_N(D) \le_P Enum(D)$.
- $N \le M \Leftrightarrow Enum_N(D) \le_P Enum_M(D)$.
- $N = |D| \Rightarrow Enum_N(D) =_P Enum(D)$.
- $Interval\,(D) \le_P MuliInterval\,(D)$.
- $MultiInterval(D) =_P Enum(D)$.
- $Enum_N(D) \le_P Mixed_N(D)$.
- $Interval(D) \le_P Mixed_N(D)$.
- Для вида недоопределённости *Struct* в работе [16] приведены следующие соотношения:
  - $Single(D_1 \times \ldots \times D_n) \le_P Single(D_1) \times \ldots \times Single(D_n)$.
  - $Interval(D_1 \times \ldots \times D_n) =_P Interval(D_1) \times \ldots \times Interval(D_n)$.
  - $Enum(D_1 \times \ldots \times D_n) \ge_P Enum(D_1) \times \ldots \times Enum(D_n)$.

На основе приведённых отношений выстраивается иерархия:

- Если $|D| = 2$, то:

- Если $|D| = N$, то:

- В общем случае:



На схемах однонаправленная стрелка "$\longleftarrow$" соответствует отношению $\leq_P$, а двунаправленная стрелка "$\longleftrightarrow$" - отношению $=_P$,

5. Заключение

В статье были рассмотрены различные способы представления значения переменных для задачи удовлетворения ограничений. Построена иерархия способов представления по полноте.

Также отмечены некоторые практически значимые свойства видов недоопределённости:

- Для логического типа данных ($|D| = 2$) все способы представления множества эквивалентны.
- Представление множества значений в виде перечисления и в виде мультиинтервала эквивалентны.
- Представления множества в виде перечисления и в виде ограниченного перечисления обладают существенно разными свойствами. То же верно для мультиинтервала и ограниченного мультиинтервала.
- Иногда применяемый на практике способ представления в виде ограниченного мультиинтервала является некорректным и его рекомендуется избегать.

Полученные результаты могут использоваться как при формулировке задач, решаемых с помощью метода недоопределённых вычислений, так и при реализации программных продуктов на основе данного метода.

Список литературы

1. Загорулько Г.Б., Сидоров В.А., Телерман В.В. и др. НеМо+: Объектно-ориентированная среда программирования в ограничениях на основе недоопределённых моделей // КИИ'98. Шестая национальная конференция с международным участием. Сборник научных трудов в трёх томах. Том I. — Пущино, 1998. — С. 524–530.
2. Нариньяни А.С. Недоопределенность в системе представления и обработки знаний // Изв. АН СССР. Техническая кибернетика. – 1986. – № 5. – С. 8 – 11.
3. Семенов А.Л. Методы распространения ограничений: основные концепции // PSI'03/ИМРО. – Интервальная математика и методы распространения ограничений, 2003.

4. Сидоров В.А. Программирование в ограничениях с чёрными ящиками. — Новосибирск, 2003. — 39 с. (Препр. / ЗАО Ледас; N2).
5. Телерман В.В. Использование мультиинтервалов в недоопределённых моделях. // Тез. докл. X всесоюз. семинара " Параллельное программирование и высокопроизводительные системы: Методы представления знаний в информационных технологиях". — Киев, 1990.
6. Телерман В.В., Сидоров В.А., Ушаков Д.М. Интервальные и мультиинтервальные расширения в недоопределённых моделях // Вычислительные технологии №1. — Т. 2. — 1997. — С. 62–70.
7. Тыугу Э.Х. Концептуальное программирование. — Москва, 1984. — 255 с.
8. Хансен Э., Уолстер Дж.У. Глобальная оптимизация с помощью методов интервального анализа: Пер. с англ. / Под ред. С. Кумков. — М.-Ижевск: Регулярная и хаотическая динамика, 2012. — 520 с.
9. Щербина О.А. Удовлетворение ограничений и программирование в ограничениях. // Интеллектуальные системы. 2011. — Т. 15, вып. 1-4. — С. 54–73.
10. Christophe Lecoutre. Constraint Networks: Techniques and Algorithms. — ISTE Ltd and John Wiley & Sons Inc, 2009. — 573 p.
11. Huffman D.A. Impossible objects as nonsence sentences // Machine Intelligence. — 1971. — Vol. 6. — P. 295–323.
12. Hyvonen E. Constraint Resoning Based on Interval Arithmetic. // Proc. IJCAI. 1989. — P. 193–199.
13. Lecoutre C., Cardon S. A greedy approach to establish singleton arc consistency // Proc. IJCAI. 2005. — P. 199–204.
14. Mackworth A.K. Consistency in Networks of Relations. // Artificial Intelligence. — 1977. — N 8. — P. 99–118.
15. Tsang E. Foundation of Constraint Satisfaction. — London: Academic Press Ltd., 1993. — 405 p.
16. Ushakov D. Some Formal Aspects of Subdifinite Models. — Novosibirsk, 1998.— 23 p. (Preprint / A. P. Ershov Institute of Informatics Systems, Siberian Division of Russian Academy of Sciences; N 49).