

ФОРМАЛЬНАЯ СПЕЦИФИКАЦИЯ АСИНХРОННЫХ ПАРАЛЛЕЛЬНЫХ ВЫЧИСЛЕНИЙ НАД ОБЩЕЙ ПАМЯТЬЮ

С.В. Панков

Южный федеральный университет, Ростов-на-Дону

Исследуется корректность асинхронных параллельных алгоритмов (всех его возможных вычислений над общей памятью) относительно заданных пред- и постусловий. Такая корректность гарантирует правильность результата для всех успешно завершающихся вычислений, началом которых является один из допустимых наборов входных данных. Демонстрируется подход к спецификации и обоснованию правильности асинхронных параллельных алгоритмов на основе формализма L-программ. Этот формализм включает в себя логико-математическую модель параллельных вычислений – L-программы (впервые описанные в [1]) и методы верификации L-программ [2].

Введение

Проблема обоснования функциональной корректности параллельных вычислительных алгоритмов является актуальной, особенно для систем, использующих асинхронные (неблокирующие) методы синхронизации параллельно исполняемых вычислительных процессов. В том числе это относится к перспективным системам с общей транзакционной памятью [3], в которых обеспечивается непротиворечивость результатов работы параллельных процессов. Семантика L-программ позволяет достаточно адекватно специфицировать вычисления в системах такого рода. Главной особенностью подхода на основе L-программ является его способность специфицировать и анализировать вычисления независимо от размера решаемой задачи [4]. Необходимо отметить, что здесь не затрагиваются такие свойства корректности, как отсутствие тупиков и завершаемость.

1. Основные понятия

Пусть $S=(Q,T)$ – система переходов, где Q – множество состояний, а $T\subseteq Q\times Q$ – отношение перехода на Q . Через $post[S](P)$ обозначим множество состояний непосредственно достижимых из $P\subseteq Q$, т.е. $post[S](P)=\{q\in Q \mid \exists q'\in Q((q',q)\in T \text{ и } q'\in P)\}$.

Множеством состояний, достижимых из $P\subseteq Q$ в системе переходов S за конечное число шагов, называется множество $post^*[S](P)=\bigcup_{i\geq 0} post^i[S](P)$, где $post^0[S](P)=P$, $post^{i+1}[S](P)=post[S](post^i[S](P))$.

Множество состояний $P\subseteq Q$ называется *инвариантом* системы переходов S , если для него справедливо $post[S](P)\subseteq P$ (войдя в инвариант, система S уже не сможет его покинуть). В соответствии с определением множества $post[S](P)$ справедливо следующее *условие инвариантности*: $\forall q',q\in Q ((q',q)\in T \ \& \ q'\in P)\Rightarrow q\in P$.

Через $fin[S]$ обозначим множество *заключительных состояний* системы переходов S , из которых нет выхода, т.е. $fin[S]=\{q\in Q \mid \forall q'\in Q((q,q')\notin T)\}$. Пусть отношение $\{I\}S\{E\}$ выражает *частичную корректность* системы переходов S относительно предусловия $I\subseteq Q$ и постусловия $E\subseteq Q$. Это отношение определяется, как: $\{I\}S\{E\} \Leftrightarrow post^*[S](I)\cap fin[S]\subseteq E$ (т.е. все заключительные состояния, достижимые системой S из предусловия I , принадлежат постусловию E).

2. *L*-программы, как частный случай систем переходов

Множество *L*-программ определяется заданием многосортного языка *L* логики предикатов первого порядка (который также является языком некоторой предметной области). Через *Q* обозначим класс всех частичных (т.е. с частичными функциями) *L*-структур. *L*-программа действует на *Q* и работает по шагам, преобразуя одну *L*-структуру в другую. В отличие от обычных логических программ, *L*-программы могут произвольным образом переопределять значения функций и предикатов. Их работа аналогична прямому выводу в логических программах и допускает одновременное исполнение разных правил и различных конкретизаций одного и того же правила.

Синтаксис *L*-программ. *L*-программа задаётся конечной совокупностью правил вида $cond(\bar{x}) \Rightarrow act(\bar{x})$, где *cond* – условие, *act* – действие, $\bar{x} = \langle x_1, \dots, x_n \rangle$ – кортеж переменных. В этом правиле $cond(\bar{x})$ – произвольная *L*-формула, $act(\bar{x})$ – формула вида $act_1(\bar{x}) \wedge \dots \wedge act_N(\bar{x})$, где $act_i(\bar{x})$ – это *L*-формула вида $p(\bar{x}_p)$, $\neg p(\bar{x}_p)$ или $f(\bar{x}_f) = y$ (*p* – предикат, *f* – функция из *L*). Причём, кортежи переменных $\bar{x}_p$, $\bar{x}_f$ состоят из переменных кортежа $\bar{x}$, *y* – переменная того же сорта, что и значение функции *f*, и *y* также из $\bar{x}$. В *L* будем различать изменяемые и неизменяемые *L*-программой функции и предикаты. Под *изменяемыми* будем понимать функции и предикаты, входящие в действия правил *L*-программы.

Перед правилом или группой правил может стоять выражение вида $\$ \bar{z}$, называемое *синхронизатором*, где $\bar{z}$ – кортеж переменных из числа переменных, входящих в условия этих правил. Группу правил с синхронизатором будем называть *синхрогруппой*.

Пример 1. В качестве примера рассмотрим задачу поиска наибольшего общего делителя произвольного числа *m* целых положительных чисел (где $m \geq 2$). Определим для этой задачи язык предметной области *L* и специфицирующую *L*-программу *НОД*. (Этот пример также предваряет следующее ниже строгое определение семантики *L*-программ).

Сорта: *Nat* – натуральные числа 1, 2, ...:

Loc – множество ячеек $\{b_1, \dots, b_m\}$, содержащих обрабатываемые числа.

Функции: $s: Loc \rightarrow Nat$ – тотальная функция, сопоставляющая каждой ячейке из *Loc*, содержащееся в ней значение:

$_-_-: Nat \times Nat \rightarrow Nat$ – обычный арифметический минус.

Предикаты: $_{<}: Nat \times Nat$ – обычное арифметическое отношение меньше.

Переменные: *u, v*: *Loc*, *h*: *Nat*.

***L*-программа *НОД*:** $s(u) < s(v) \wedge h = s(v) - s(u) \Rightarrow s(v) = h$.

В соответствии с определяемой ниже семантикой *L*-программ, на каждом шаге работы *НОД* правило выполняется для некоторых пар ячеек *u* и *v*, для которых справедливо условие $s(u) < s(v)$. При этом, содержимое ячейки *v* (большее значение) заменяется на $s(v) - s(u)$ (разность большего и меньшего значений). Пары ячеек, с которыми это происходит одновременно, выбираются недетерминированно, но с таким условием, чтобы не делалось попытки записать в одну ячейку разные значения (чтобы не выполнялись противоречивые действия).

Синхронизатор уменьшает степень недетерминизма *L*-программы. Так, например, в *L*-программе *НОД* недетерминизм можно ограничить следующим образом: $\$ v, h$ $s(u) < s(v) \wedge h = s(v) - s(u) \Rightarrow s(v) = h$. Благодаря синхронизатору $\$ v, h$, если на некотором шаге работы этой *L*-программы правило выполняется для пары ячеек $u = b_i$ и $v = b_j$ (где $1 \leq i \leq m$, $1 \leq j \leq m$), то на этом же шаге оно исполнится и для всех пар ячеек $u = b_i$ и $v = b_k$ (где $1 \leq k \leq m$), для которых справедливо $s(b_i) < s(b_k)$. Одна ячейка с меньшим значением и все ячейки с большими (чем в первой) значениями, составляют группу пар ячеек, для которых одновременно выполняется правило. Недетерминированность остаётся на уровне

выбора ячейки с меньшим значением. Одновременное исполнение правила для нескольких таких групп пар ячеек невозможно, когда содержимое ячеек с меньшими значениями для этих групп не совпадает (из-за наличия в этом случае противоречивых действий).

Операционная семантика L -программ. Определим операционную семантику L -программ, но сначала введем некоторые понятия и обозначения. Пусть q – L -структуры из Q , C_q – множество констант для обозначения объектов L -структуры q . Пусть $r(\bar{x})$ обозначает правило L -программы $cond(\bar{x}) \Rightarrow act(\bar{x})$. Ансамблем на q , порождаемым правилом $r(\bar{x})$, назовём произвольную конкретизацию этого правила на q , т.е. запись вида $r(\bar{c})$, где $\bar{c} = \langle c_1, \dots, c_n \rangle$ ($c_i \in C_q$ для всех $1 \leq i \leq n$). Ансамбль $r(\bar{c})$ называется *активным* на q , если на q истинна формула $cond(\bar{c})$.

Пусть правило $r(\bar{z}, \bar{v})$ входит в группу с синхронизатором $\$ \bar{z}$ и на q активен ансамбль $r(\bar{a}, \bar{c})$, где $\bar{a}$ и $\bar{c}$ – конкретизация соответствующих наборов переменных $\bar{z}$ и $\bar{v}$. *Синхрорасширением* активного ансамбля $r(\bar{a}, \bar{c})$ назовём множество ансамблей $\{r(\bar{a}', \bar{c}) \mid r(\bar{a}', \bar{c}) \text{ активен на } q\}$, где $\bar{a}'$ – произвольная конкретизация набора переменных $\bar{z}$.

Атомарными действиями назовём конъюнктивные члены формулы $act(\bar{c})$, т.е. формулы вида $p(\bar{c}_p)$, $p(\bar{c}_p)$, или $f(\bar{c}_f)=b$ (здесь $\bar{c}_p$, $\bar{c}_f$ и b – конкретизация набора переменных $\bar{x}_p$, $\bar{x}_f$ и y соответственно). Активность ансамбля $cond(\bar{c}) \Rightarrow act(\bar{c})$ может привести к одновременному выполнению всех его атомарных действий. При *выполнении* на q атомарное действие $p(\bar{c}_p)$ устанавливает истинным p на q , $p(\bar{c}_p)$ устанавливает ложным p на q , а $f(\bar{c}_f)=b$ устанавливает значение f на $\bar{c}_f$ равным b .

Коалицией на q будем называть произвольное непустое множество активных ансамблей M , удовлетворяющее следующим трём условиям:

Условие непротиворечивости: объединение всех атомарных действий коалиции образует непротиворечивую совокупность формул.

Первое условие синхронизации: если в M входит ансамбль, порожденный некоторым правилом синхрогруппы, то и для любого другого правила этой синхрогруппы в M должен входить, хотя бы один, порожденный этим правилом, активный ансамбль (если такой существует).

Второе условие синхронизации: если в M входит ансамбль, порожденный правилом синхрогруппы, то в M также должно входить всё его синхрорасширение.

Исполнение коалиции состоит в выполнении всех её атомарных действий. *Шаг работы L -программы* на заданной L -структуре состоит в исполнении на ней некоторой коалиции. При этом, коалиция выбирается недетерминировано. Если на q нет ни одной коалиции, L -программа завершает свою работу в q . Таким образом, L -программа задаёт систему переходов $S=(Q, T)$, где Q – произвольное множество L -структур, замкнутое относительно шага работы L -программы, а отношение перехода $T=\{(q, q') \mid q, q' \in Q \text{ и } q' \text{ получается из } q \text{ исполнением некоторой коалиции}\}$.

3. Анализ функциональной корректности L -программ

Частичная корректность системы переходов S относительно предусловия $I \subseteq Q$ и постусловия $E \subseteq Q$ характеризует правильность системы с точки зрения вычисляемой ею функции. Допустимые входные данные функции описываются предусловием I , а соответствующий этим данным результат вычисления функции должен удовлетворять требованиям постусловия E . Установить $\{I\}S\{E\}$ можно путём проверки выполнения следующих *условий корректности* [2]: Существует такое множество состояний $W \subseteq Q$, что справедливо:

1. $I \subseteq W$ (условие инициализации);
2. $\forall q', q \in Q ((q', q) \in T \ \& \ q' \in W) \Rightarrow q \in W$ (условие инвариантности W);

3. $W \cap \text{fin}[S] \subseteq E$ (условие завершения).

Алгоритмы из [2] позволяют получить логические формулы, выражающие в языке L отношение перехода T , множества $\text{post}[S](P)$ и $\text{fin}[S]$ для L -программ. Если множества I , W и E также выражаются L -формулами, то анализ функциональной корректности сводится к доказательству истинности формул логики предикатов первого порядка.

Пример 2. Обоснуем правильность *НОД* относительно заданных пред- и постусловий. Пусть определённый в примере 1 язык L также содержит: все обычные арифметические функции и отношения; предикат $\text{делит}_\mathbf{D} : \text{Nat} \times \text{Nat}$ – выражающий тот факт, что первый аргумент является делителем второго; функция $s' : \text{Loc} \rightarrow \text{Nat}$ – имеет тот же смысл, что и функция s , только представляет содержимое ячеек из Loc в L -структуре q' ; переменная $\mathbf{d}$ сорта Nat . Знаками $\wedge, \vee, \neg, \rightarrow, \forall$ и $\exists$ будем обозначать обычные логические связки и кванторы.

Пусть $W(\mathbf{D})$ обозначает формулу, выражающую тот факт, что константа $\mathbf{D}$ (сорта Nat) является наибольшим общим делителем (н.о.д.) чисел $s(b_1), \dots, s(b_m)$

$$W(\mathbf{D}): \forall u(\mathbf{D} \text{ делит } s(u)) \wedge \forall d(\forall u(d \text{ делит } s(u)) \rightarrow d \leq \mathbf{D}).$$

Эта формула утверждает, что $\mathbf{D}$ является делителем чисел $s(b_1), \dots, s(b_m)$ и для любого другого делителя $\mathbf{d}$ этих чисел справедливо $\mathbf{d} \leq \mathbf{D}$. Формулу $W(\mathbf{D})$ будем рассматривать как кандидат в инварианты, а также примем её в качестве предусловия I . В качестве постусловия E выберем формулу $\forall u(s(u) = \mathbf{D})$, требующую, чтобы содержимое каждой ячейки совпадало с н.о.д.

Формула FIN , выражающая множество заключительных состояний $\text{fin}[S]$, утверждает, что условие правила L -программы *НОД* должно быть ложно для любой конкретизации её свободных переменных, и в соответствии с алгоритмом из [2], строится, как: $\forall u, v, h \neg (s(u) < s(v) \wedge h = s(v) - s(u))$. С учётом тотальности s на $\mathcal{Q}$ эта формула эквивалентна следующей формуле:

$$FIN: \forall u, v (s(u) = s(v)),$$

которая говорит о том, что содержимое всех ячеек должно совпадать. Формула $T(s', s)$, описывающая связь L -структур q' и q таких, что $(q', q) \in T$ (выражающая отношение перехода), в соответствии с алгоритмом из [2], имеет следующий вид:

$$T(s', s): \forall v (s(v) = s'(v) \vee \exists u (s'(u) < s'(v) \wedge s(v) = s'(v) - s'(u))).$$

Эта формула утверждает, что при переходе из q' в q содержимое ячейки, либо не изменилось, либо уменьшилось на значение некоторой другой ячейки, содержимое которой меньше значения текущей ячейки. Помимо приведённых формул, будем использовать следующее ниже свойство рассматриваемой предметной области:

$$\forall d, e_1, \dots, e_k \in \text{Nat} (e = e_1 \pm \dots \pm e_k \wedge d \text{ делит } e_i, 1 \leq i \leq k \Rightarrow d \text{ делит } e). \quad (1)$$

Доказательство. Пусть $W \langle s' \rangle (\mathbf{D})$ – это формула $W(\mathbf{D})$, в которой вхождения функции s заменены на s' . В соответствии с приведёнными выше теоретико-множественными условиями корректности, для обоснования $\{I\} \text{НОД} \{E\}$ достаточно доказать истинность следующих L -формул:

1. $I \rightarrow W(\mathbf{D})$ (условие инициализации *НОД*);
2. $T(s', s) \wedge W \langle s' \rangle (\mathbf{D}) \rightarrow W(\mathbf{D})$ (условие инвариантности $W(\mathbf{D})$ для *НОД*);
3. $W(\mathbf{D}) \wedge FIN \rightarrow E$ (условие завершения *НОД*).

Условие инициализации истинно, т.к. I совпадает с $W(\mathbf{D})$. Условие инвариантности утверждает следующее: если функции s' и s связаны так, как это описывается формулой $T(s', s)$, и $\mathbf{D}$ – н.о.д. чисел $s'(b_1), \dots, s'(b_m)$, то $\mathbf{D}$ – н.о.д. чисел $s(b_1), \dots, s(b_m)$. Доказательство истинности условия инвариантности разобьём на две части.

1. Сначала докажем истинность формулы $T(s', s) \wedge W \langle s' \rangle (\mathbf{D}) \rightarrow \forall u (\mathbf{D} \text{ делит } s(u))$. Из $T(s', s)$ следует, что для произвольного $b_i \in \text{Loc}$ либо $s(b_i) = s'(b_i)$, либо $s(b_i) = s'(b_i) - s'(b_j)$ для некоторого $b_j \in \text{Loc}$. Из $W \langle s' \rangle (\mathbf{D})$ следует $\forall u (\mathbf{D} \text{ делит } s'(u))$, тогда по свойству

- (1) справедливо $D \text{ делит } s(b_i)$. В силу произвольности b_i , истинна формула $\forall u(D \text{ делит } s(u))$.
2. Докажем истинность формулы $T(s',s) \wedge W<s'>(D) \rightarrow \forall d(\forall u(d \text{ делит } s(u)) \rightarrow d \leq D)$. Она эквивалентна формуле $\exists d(\forall u(d \text{ делит } s(u)) \wedge d > D) \wedge T(s',s) \rightarrow \neg W<s'>(D)$. Из $T(s',s)$ следует, что для произвольного $b_{i0} \in Loc$ либо $s'(b_{i0}) = s(b_{i0})$, либо $s'(b_{i0}) = s(b_{i0}) + s'(b_{i1})$ для некоторого $b_{i1} \in Loc$ такого, что $s'(b_{i1}) < s'(b_{i0})$. Во втором варианте $s'(b_{i1})$ выражается по формуле $T(s',s)$ через функцию s аналогично $s'(b_{i0})$ и т.д., а в силу конечности Loc существует n такое, что $s'(b_{i0}) = s(b_{i0}) + s(b_{i1}) + \dots + s(b_{in})$. Тогда в силу свойства (1) и произвольности b_{i0} из формулы $\exists d(\forall u(d \text{ делит } s(u)) \wedge d > D)$ следует истинность формулы $\exists d(\forall u(d \text{ делит } s'(u)) \wedge d > D)$. Тогда истинна формула $W<s'>(D)$, т.к. она эквивалентна формуле $\exists u(D \text{ делит } s'(u)) \vee \exists d(\forall u(d \text{ делит } s''(u)) \wedge d > D)$. Таким образом, истинность условия инвариантности доказана.
- Истинность условия завершения $W(D) \wedge \forall u, v(s(u) = s(v)) \rightarrow \forall u(s(u) = D)$ по свойству н.о.д. очевидна. Тем самым доказана частичная корректность L -программы $НОД$.

Литература

1. Крицкий С.П. Модель асинхронных вычислений в структурах и языки программирования // Методы трансляции. Ростов-на-Дону, 1981. С. 92-100.
2. Крицкий С.П., Панков С.В. О верификации асинхронных программ продукционного типа // Программирование. № 5, 1994, С. 40-52.
3. Grahn H. Transactional Memory // J. Parallel Distrib. Comput. 2010. Vol. 70 (10). P. 993-1008.
4. Панков С.В. Логический подход к спецификации и анализу поведения клеточных автоматов // Труды Всероссийской научной конференции «Научный сервис в сети Интернет: многоядерный компьютерный мир. 15 лет РФФИ», Новороссийск, 19-24 сентября 2007 г. Изд-во Московского университета. С. 85-88.