Наука и Образование
МГТУ им. Н.Э. Баумана
Сетевое научное издание
ISSN 1994-0448

Наука и Образование. МГТУ им. Н.Э. Баумана.
Электрон. журн. 2014. № 11. С. 535–554.

DOI: **10.7463/1114.0741563**

Представлена в редакцию: 23.11.2014
Исправлена: 29.11.2014

УДК 004.051+519.168

Оценка эффективности параллельных алгоритмов операций преобразования графовой модели

Иванова Г. С.[1,*], Головков А. А.[1]

[*] gsivanova@gmail.com

[1]МГТУ им. Н.Э. Баумана, Москва, Россия

При решении задач на графах больших размерностей возникает необходимость применения высокопроизводительных вычислительных систем, так как выполнение большого количества операций над графами требует значительного времени. Для увеличения быстродействия используются разные методы, в том числе применение параллельных вычислительных систем. Целью статьи является выявление элементарных операций, из которых состоят операции над графом, построение параллельных алгоритмов операций над графовыми моделями с использованием выявленных элементарных операций, оценка эффективности предложенной реализации. Анализ операций преобразования графа, наиболее часто встречающихся в алгоритмах решения задач, выполнен независимо от особенностей архитектуры конкретных параллельных систем, что позволит в дальнейшем использовать параллельные операции на различных системах параллельной обработки. Полученная оценка «снизу» коэффициентов ускорения для различных операций показала высокую эффективность применения параллельных алгоритмов операций над графами.

Ключевые слова: графы, операции над графами, параллельные алгоритмы, параллельные вычислительные системы, коэффициент ускорения

Введение

В связи с появлением новых задач больших размерностей, методы решения которых базируются на теории графов [7-9], возникает необходимость выявления новых способов обработки графовых моделей [10-12].

Одно из основных направлений при исследовании и разработки систем, работающих с графами, – использование параллельных вычислительных систем [6,13].

Так параллельные вычислительные системы могут успешно применяться при решении NP-полных задач над графами, решение которых требует значительных вычислительных мощностей, так как обработка графов представляет собой преобразование и анализ отдельных структур, составляющих граф, и может быть эффективно реализована в параллельных вычислительных системах.

В связи с этим возникает задача анализа операций над графами с точки зрения их реализации в системе параллельной обработки.

Условные обозначения:

$Op_{AddEl}(E, e) = E.e$ – операция добавления элемента e во множество E;

$Op_{AddSet}(A, B) = A.B$ – операция добавления множества B во множество множеств A;

$Op_{DeleteEl}(E, e) = E \backslash e$ – операция удаления элемента e из множества E;

$Op_{DeleteSet}(A, B) = A \backslash B$ – операция удаления множества B из множества множеств A;

$Op_{UnionSet}(Z, X, Y)$, где $Z = X \cup Y$ – операция объединения множеств X и Y с сохранением результата во множестве Z.

1. Основные положения и допущения

С учетом большой размерности решаемых задач будем представлять граф G аналитически, т.е. множествами вершин X, $|X| = n$, ребер U, $|U| = m$ и множествами множеств его образов по отношениям инцидентности $\Gamma_1 X$ и $\Gamma_2 U$. Этого достаточно для описания любой графовой модели (ориентированных и неориентированных графов, мультиграфов, гиперграфов и ультраграфов) [2-5].

Преобразование и анализ графов есть выполнение над графами различных операций, результатом которых может являться модифицированный граф, какая-либо характеристика графа, новый граф, полученный из старого и т.д. [3-5].

С учетом представления графа множествами и множествами множеств каждая операция над графом реализуется как некоторая совокупность операций над множествами, описывающими граф, например, операций добавления элементов во множества и/или удаления элементов из них.

Большая часть таких элементарных операций выполняется над разными множествами, следовательно, эти операции могут быть выполнены параллельно, что значительно уменьшит время выполнения операции над графом в целом.

Оценим эффективность использования параллельных алгоритмов выполнения операций над графами исходя из абстрактного описания графа множествами, которые могут быть реализованы в виде любых структур [1,6], удобных для обработки в конкретной параллельной вычислительной системе.

Так же для ограничения уровня анализа и упрощения модели параллельных операций положим, что любая элементарная операция преобразования или просмотра множества является единицей выполнения потока. То есть один поток за определенное время способен выполнить только одну элементарную операцию над одним множеством. При этом потоки, взаимодействующие с одинаковыми множествами, не могут выполняться параллельно, так как зависимы по данным.

Указанные требования накладывают достаточно жесткие ограничения на параллельные реализации операций над графами, снижая их эффективность, так как при выборе конкретной структуры данных теоретически возможно распараллеливание самих операций над этой структурой. Выбор структуры данных для представления графа в параллельной вычислительной системе, обеспечивающих максимальную степень параллелизма при выполнении операций над графом, – тема отдельного исследования и в настоящей работе затрагиваться не будет.

2. Оценка эффективности выполнения операций над графами параллельными системами

В [2-5] приводятся основные операции преобразования графов:

1) добавление вершины;
2) добавление ребра;
3) удаление вершины;
4) удаление ребра;
5) свертка множества вершин графа;
6) стягивание ребер графа;
7) декомпозиция вершины графа;
8) подразбиение ребра графа;
9) удаление вершины из образов и прообразов инцидентных ей ребер;
10) удаление ребра из образов и прообразов инцидентных ему вершин;
11) формирование части графа;
12) дополнение части до графа или его куска;
13) объединение графов;
14) пересечение графов;
15) кольцевая сумма графов;
16) композиция графов;
17) произведение графов;
18) суперпозиция графов;
19) соединение графов.

В работе будем рассматривать операции 1 – 10, так как, во-первых, они наиболее часто встречаются в алгоритмах решения задач на графах, во-вторых, эти операции являются базовыми преобразованиями графа, остальные операции во многом состоят из них, в-третьих, операции 1 – 10 могут быть удобно представлены в виде совокупности элементарных операций без использования ветвлений и циклов, в то время как операции 11 – 19 целесообразно описывать в виде алгоритма.

В процессе анализа для каждой операции будем выявлять элементарные операции, строить ее параллельный алгоритм, находить время выполнения последовательного и параллельного алгоритма и коэффициент ускорения.

2.1. Добавление вершины

Исходными данными для операции добавления вершины являются:

1) x_k – добавляемая вершина;
2) U' – множество входящих ребер, которым инцидентна вершина x_k, $0 \le |U'| \le m$;
3) U'' – множество исходящих ребер, которые инцидентны вершине x_k, $0 \le |U''| \le$ $\le m$.

Реализация операции включает:

- добавление вершины x_k во множество X вершин графа:

$$X.x_k;$$

- множество операций добавления вершины x_k в образы инцидентных ей ребер:

$$\{\Gamma_2 U_i'.x_k, i = \overline{1,p}, p = |U'|\};$$

- множество операций добавления ребер $u_j \ni U''$ в образ $\Gamma_1 x_k$:

$$\{\Gamma_1 x_k.u_j \ni U'', j = \overline{1,q}, q = |U''|\}.$$

Добавление элемента во множество во всех случаях предполагает, что ранее такой вершины в перечисленных выше множествах не было, поэтому вместо операции объединения элемента e и множества E следует использовать операцию конкатенации:

$$Op_{AddEl}(E,e) = E.e. \tag{1}$$

Указанные выше операции добавления вершины x_k выполняются с различными множествами. Следовательно, их можно выполнять параллельно. Операции добавления ребер $u_j \ni U''$ выполняются с одним множеством, а потому при принятых ограничениях распараллеливанию не подлежат.

Для наглядности отдельные элементарные операции добавления элемента или их последовательности, представляющие собой нити, которые могут выполняться независимо от других, запишем в отдельных строках. Тогда параллельная реализация операции добавления вершины $P_{AddVertex}(G, x_k, U', U'')$ может быть описана следующим образом:

$$P_{AddVertex}(G, x_k, U', U'') => \left[\begin{array}{l} Op_{AddEl}(X, x_k), \\ \{Op_{AddEl}(\Gamma_2 U_1', x_k), \\ \quad \ldots \\ Op_{AddEl}(\Gamma_2 U_p', x_k), p = |U'|\}, \\ \{Op_{AddEl}(\Gamma_1 x_k, u_j \ni U''), j = \overline{1,q}, q = |U''|\}. \end{array}\right.$$

Учтем также, что множество Γx_k в описании исходном графа отсутствовало, и, следовательно, прежде чем формировать это множество его необходимо создать. Поэтому вводится элементарная операция добавления нового множества во множество множеств:

$$Op_{AddSet}(\Gamma_1 X, \Gamma_1 x_k) = \Gamma_1 X.\Gamma_1 x_k. \tag{2}$$

Тогда параллельная реализация операции принимает вид:

$$P_{AddVertex}(G, x_k, U', U'') =>$$

$$=> \left[\begin{array}{l} Op_{AddEl}(X, x_k), \\ \{Op_{AddEl}(\Gamma_2 U_1', x_k), \\ \quad \ldots \\ Op_{AddEl}(\Gamma_2 U_p', x_k), p = |U'|\}, \\ Op_{AddSet}(\Gamma_1 X, \Gamma_1 x_k), \{Op_{AddEl}(\Gamma_1 x_k, u_j \ni U''), j = \overline{1,q}, q = |U''|\}. \end{array}\right.$$

В соответствии с принятым обозначением последовательности операций, указанные в отдельных строках, могут выполняться параллельно.

Рассчитаем коэффициент ускорения при использовании параллельной реализации операции.

При $t(Op_{AddEl}) \cong t(Op_{AddSet})$, где $t(Op)$ – время выполнения операции $Op,$ получаем:

- время выполнения параллельной реализации операции совпадает со временем выполнения самой длинной цепочки:

$$t(P_{AddVertex}) = t((1+|U''|)*Op_{AddEl}) = (1+|U''|)*t(Op_{AddEl});$$

- время выполнения последовательной реализации операции:

$$t(\overline{P_{AddVertex}}) = t\big((2+|U'|+|U''|)*Op_{AddEl}\big) = (2+|U'|+|U''|)*t(Op_{AddEl}).$$

Откуда коэффициент ускорения:

$$K_{AddVertex} = \frac{t(\overline{P_{AddVertex}})}{t(P_{AddVertex})} = \frac{(2+|U'|+|U''|)*t(Op_{AddEl})}{(1+|U''|)*t(Op_{AddEl})} = \frac{2+|U'|+|U''|}{1+|U''|}.$$

Следовательно:

- в лучшем случае при $|U'| = m, |U''| = 0$: $K^{max}_{AddVertex} = 2+m$;
- в худшем случае при $|U'| = 0, |U''| = m$: $K^{min}_{AddVertex} = \frac{2+m}{1+m}$.

2.2. Добавление ребра

С точки зрения реализации операция аналогична операции, рассмотренной в п. 2.1. Входными данными являются:

1) u_k – добавляемое ребро;
2) X' – множество вершин, которые инцидентны ребру $u_k, 0 \le |X'| \le n$;
3) X''– множество вершин, которым инцидентно ребро $u_k, 0 \le |X''| \le n$.

Тогда параллельная реализация операции принимает вид:

$$P_{AddEdge}(G, u_k, X', X'') =>$$

$$=> \begin{cases} Op_{AddEl}(U, u_k), \\ \{Op_{AddEl}(\Gamma_1 X'_1, u_k), \\ \dots \\ Op_{AddEl}\big(\Gamma_1 X'_p, u_k\big), p = |X'|\}, \\ Op_{AddSet}(\Gamma_2 U, \Gamma_2 u_k), \{Op_{AddEl}\big(\Gamma_2 u_k, x_j \ni X''\big), j = \overline{1,q}, q = |X''|\}. \end{cases}$$

Рассчитаем коэффициент ускорения при использовании параллельной реализации операции.

При $t(Op_{AddEl}) \cong t(Op_{AddSet})$:

- время выполнения параллельной операции:

$$t\big(P_{AddEdge}\big) = t((1+|X''|*Op_{AddEl}) = (1+|X''|)*t(Op_{AddEl});$$

- время выполнения последовательной операции:

$$t\big(\overline{P_{AddEdge}}\big) = t\big((2+|X'|+|X''|)*Op_{AddEl}\big) = (2+|X'|+|X''|)*t(Op_{AddEl}).$$

Откуда коэффициент ускорения:

$$K_{AddEdge} = \frac{t\left(\overline{P_{AddEdge}}\right)}{t\left(P_{AddEdge}\right)} = \frac{(2 + |X'| + |X''|) * t(Op_{AddEl})}{(1 + |X''|) * t(Op_{AddEl})} = \frac{2 + |X'| + |X''|}{1 + |X''|}.$$

Следовательно:

- в лучшем случае при $|X'| = n, |X''| = 0$: $K_{AddEdge}^{max} = 2 + n$;
- в худшем случае при $|X'| = 0, |X''| = n$: $K_{AddEdge}^{min} = \frac{2+n}{1+n}$.

2.3. Удаление вершины из графа

Для операции входными данными является x_k – удаляемая вершина.

Реализация операции предполагает:

- удаление вершины x_k из множества X вершин графа:

$$X \backslash x_k;$$

- удаление образа $\Gamma_1 x_k$ из отображения $\Gamma_1 X$:

$$\Gamma_1 X \backslash \Gamma_1 x_k;$$

- множество операций удаления вершины x_k из образов инцидентных ей ребер:

$$\{\Gamma_2 U_i \backslash x_k, i = \overline{1, m}\}.$$

В соответствии с этим вводятся элементарные операции удаления элемента из множества и удаления множества из множества множеств:

$$Op_{DeleteEl}(E, e) = E \backslash e; \quad (3)$$

$$Op_{DeleteSet}(A, B) = A \backslash B. \quad (4)$$

Тогда параллельная реализация операции принимает вид:

$$P_{DeleteVertex}(G, x_k) => \begin{cases} Op_{DeleteEl}(X, x_k), \\ Op_{DeleteSet}(\Gamma_1 X, \Gamma_1 x_k), \\ \{Op_{DeleteEl}(\Gamma_2 U_1, x_k), \\ \ldots \\ Op_{DeleteEl}(\Gamma_2 U_m, x_k)\}. \end{cases}$$

Рассчитаем коэффициент ускорения при использовании параллельной реализации операции.

При $t(Op_{DeleteEl}) \cong t(Op_{DeleteSet})$:

- время выполнения параллельной операции:

$$t(P_{DeleteVertex}) = t(Op_{DeleteEl});$$

- время выполнения последовательной операции:

$$t(\overline{P_{DeleteVertex}}) = t\left((2 + m) * Op_{DeleteEl}\right) = (2 + m) * t(Op_{DeleteEl}).$$

Откуда коэффициент ускорения:

$$K_{DeleteVertex} = \frac{t(\overline{P_{DeleteVertex}})}{t(P_{DeleteVertex})} = \frac{(2 + m) * t(Op_{DeleteEl})}{t(Op_{DeleteEl})} = 2 + m.$$

Следовательно, коэффициент ускорения и в лучшем, и в худшем случае: $K_{DeleteVertex} = 2 + m$.

2.4. Удаление ребра из графа

С точки зрения реализации операция аналогична операции, рассмотренной в п. 2.3. Входными данными является u_k – удаляемое ребро.

Тогда параллельная реализация операции принимает вид:

$$P_{DeleteEdge}(G, u_k) => \begin{cases} Op_{DeleteEl}(U, u_k), \\ Op_{DeleteSet}(\Gamma_2 U, \Gamma u_k), \\ \{Op_{DeleteEl}(\Gamma_1 X_1, u_k), \\ \dots \\ Op_{DeleteEl}(\Gamma_1 X_n, u_k)\}. \end{cases}$$

Рассчитаем коэффициент ускорения при использовании параллельной реализации операции.

При $t(Op_{DeleteEl}) \cong t(Op_{DeleteSet})$:

- время выполнения параллельной операции:

$$t\left(P_{DeleteEdge}\right) = t(Op_{DeleteEl});$$

- время выполнения последовательной операции:

$$t\left(\overline{P_{DeleteEdge}}\right) = t\left((2+n) * Op_{DeleteEl}\right) = (2+n) * t(Op_{DeleteEl}).$$

Откуда коэффициент ускорения:

$$K_{DeleteEdge} = \frac{t\left(\overline{P_{DeleteEdge}}\right)}{t\left(P_{DeleteEdge}\right)} = \frac{(2+n) * t(Op_{DeleteEl})}{t(Op_{DeleteEl})} = 2 + n.$$

Следовательно, коэффициент ускорения и в лучшем, и в худшем случае:

$$K_{DeleteEdge} == 2 + n.$$

2.5. Свертка множества вершин графа

Для операции входными данными являются x_f, x_k – свертываемые вершины.

Реализация операции предполагает:

- удаление вершины x_k из множества X вершин графа:

$$X \backslash x_k;$$

- объединение образов $\Gamma_1 x_f$ и $\Gamma_1 x_k$:

$$\Gamma_1 x_f \cup \Gamma_1 x_k;$$

- множество операций удаления вершины x_k из образов инцидентных ей ребер:

$$\{\Gamma_2 U_i \backslash x_k, i = \overline{1, m}\};$$

- удаление образа $\Gamma_1 x_k$ из отображения $\Gamma_1 X$:

$$\Gamma_1 X \backslash \Gamma_1 x_k.$$

В соответствии с этим вводится элементарная операция объединения множеств:

$$Op_{UnionSet}(Z, X, Y), \text{где } Z = X \cup Y. \quad (5)$$

Тогда реализация параллельной операции принимает вид:

$$P_{UnionVertices}(G, X_f, X_k) => \begin{cases} Op_{DeleteEl}(X, x_k), \\ Op_{UnionSet}(\Gamma_1 x_f, \Gamma_1 x_f, \Gamma_1 x_k), Op_{DeleteSet}(\Gamma_1 X, \Gamma_1 x_k), \\ \{Op_{DeleteEl}(\Gamma_2 U_1, x_k), \\ \dots \\ Op_{DeleteEl}(\Gamma_2 U_m, x_k)\}. \end{cases}$$

Рассчитаем коэффициент ускорения при использовании параллельной реализации операции.

При $t(Op_{DeleteEl}) \cong t(Op_{DeleteSet}) \cong t(Op_{AddEl})$ и $t(Op_{UnionSet}) \cong t(|\Gamma x_k| * |\Gamma x_f| * * Op_{AddEl})$:

- время выполнения параллельной операции:

$$t(P_{UnionVertices}) = \begin{cases} t\left(\left(|\Gamma_1 x_k| * |\Gamma_1 x_f|\right) * Op_{AddEl}\right) + t(Op_{AddEl}), \text{при } |\Gamma_1 x_k| * |\Gamma_1 x_f| > 1 \\ t(Op_{AddEl}) + t(Op_{AddEl}), \text{иначе} \end{cases} =$$

$$= \begin{cases} \left(1 + |\Gamma_1 x_k| * |\Gamma_1 x_f|\right) * t(Op_{AddEl}), \text{при } |\Gamma_1 x_k| * |\Gamma_1 x_f| > 1 \\ 2 * t(Op_{AddEl}), \text{иначе} \end{cases};$$

- время выполнения последовательной операции:

$$t(\overline{P_{UnionVertices}}) = t\left(\left(1 + |\Gamma_1 x_k| * |\Gamma_1 x_f| + m + 1\right) * Op_{AddEl}\right) =$$

$$= \left(2 + m + |\Gamma_1 x_k| * |\Gamma_1 x_f|\right) * t(Op_{DeleteEl}).$$

Откуда коэффициент ускорения:

$$K_{UnionVertices} = \frac{t(\overline{P_{UnionVertices}})}{t(P_{UnionVertices})} =$$

$$= \begin{cases} \dfrac{\left(2 + m + |\Gamma_1 x_k| * |\Gamma_1 x_f|\right) * t(Op_{DeleteEl})}{\left(1 + |\Gamma_1 x_k| * |\Gamma_1 x_f|\right) * t(Op_{AddEl})}, \text{при } |\Gamma_1 x_k| * |\Gamma_1 x_f| > 1 \\ \dfrac{\left(2 + m + |\Gamma_1 x_k| * |\Gamma_1 x_f|\right) * t(Op_{DeleteEl})}{2 * t(Op_{AddEl})}, \text{иначе} \end{cases} =$$

$$= \begin{cases} \dfrac{2 + m + |\Gamma_1 x_k| * |\Gamma_1 x_f|}{1 + |\Gamma_1 x_k| * |\Gamma_1 x_f|}, \text{при } |\Gamma_1 x_k| * |\Gamma_1 x_f| > 1 \\ \dfrac{2 + m + |\Gamma_1 x_k| * |\Gamma_1 x_f|}{2}, \text{иначе} \end{cases} =$$

$$= \begin{cases} 1 + \dfrac{1 + m}{1 + |\Gamma_1 x_k| * |\Gamma_1 X_f|}, \text{при } |\Gamma x_k| * |\Gamma_1 x_f| > 1 \\ 1 + \dfrac{m + |\Gamma_1 x_k| * |\Gamma_1 x_f|}{2}, \text{иначе} \end{cases}.$$

Следовательно:

- в лучшем случае при $|\Gamma_1 x_k| * |\Gamma_1 x_f| = 1$: $K^{max}_{UnionVertices} = 1 + \frac{1+m}{2}$;
- в худшем случае при $|\Gamma_1 x_k| * |\Gamma_1 x_f| = 0, m = 1$: $K^{min}_{UnionVertices} = 1.5$.

2.6. Стягивание ребер графа

С точки зрения реализации операция аналогична п. 2.5. Входными данными являются u_f, u_k – стягиваемые ребра.

Тогда реализация параллельной операции принимает вид:

$$P_{UnionEdges}(G, U_f, U_k) => \begin{bmatrix} Op_{DeleteEl}(U, u_k), \\ Op_{UnionSet}(\Gamma_2 u_f, \Gamma_2 u_f, \Gamma_2 u_k), Op_{DeleteSet}(\Gamma_2 U, \Gamma_2 u_k), \\ \{Op_{DeleteEl}(\Gamma_1 X_1, U_k), \\ \dots \\ Op_{DeleteEl}(\Gamma_1 X_n, u_k)\}. \end{bmatrix}$$

Рассчитаем коэффициент ускорения при использовании параллельной реализации операции.

При $t(Op_{DeleteEl}) \cong t(Op_{DeleteSet}) \cong t(Op_{AddEl})$ и $t(Op_{UnionSet}) \cong t(|\Gamma_2 u_k| * |\Gamma_2 u_f| * * Op_{AddEl})$:

- время выполнения параллельной реализации операции:

$$t(P_{UnionEdges}) = \begin{cases} t\left((|\Gamma_2 u_k| * |\Gamma_2 u_f|) * Op_{AddEl}\right) + t(Op_{AddEl}), \text{при } |\Gamma_2 u_k| * |\Gamma_2 u_f| > 1 \\ t(Op_{AddEl}) + t(Op_{AddEl}), \text{иначе} \end{cases} =$$

$$= \begin{cases} (1 + |\Gamma_2 u_k| * |\Gamma_2 u_f|) * t(Op_{AddEl}), \text{при } |\Gamma_2 u_k| * |\Gamma_2 u_f| > 1 \\ 2 * t(Op_{AddEl}), \text{иначе} \end{cases};$$

- время выполнения последовательной реализации операции:

$$t(\overline{P_{UnionEdges}}) = t\left((1 + |\Gamma_2 u_k| * |\Gamma_2 u_f| + n + 1) * Op_{AddEl}\right) =$$
$$= (2 + n + |\Gamma_2 u_k| * |\Gamma_2 u_f|) * t(Op_{DeleteEl}).$$

Откуда коэффициент ускорения:

$$K_{UnionEdges} = \frac{t(\overline{P_{UnionEdges}})}{t(P_{UnionEdges})} =$$

$$= \begin{cases} \dfrac{(2 + n + |\Gamma_2 u_k| * |\Gamma_2 u_f|) * t(Op_{DeleteEl})}{(1 + |\Gamma_2 u_k| * |\Gamma_2 u_f|) * t(Op_{AddEl})}, \text{при } |\Gamma_2 u_k| * |\Gamma_2 u_f| > 1 \\ \dfrac{(2 + n + |\Gamma_2 u_k| * |\Gamma_2 u_f|) * t(Op_{DeleteEl})}{2 * t(Op_{AddEl})}, \text{иначе} \end{cases} =$$

$$= \begin{cases} \dfrac{2 + n + |\Gamma_2 u_k| * |\Gamma_2 u_f|}{1 + |\Gamma_2 u_k| * |\Gamma_2 u_f|}, \text{при } |\Gamma_2 u_k| * |\Gamma_2 u_f| > 1 \\ \dfrac{2 + n + |\Gamma_2 u_k|}{2}, \text{иначе} \end{cases} =$$

$$= \begin{cases} 1 + \dfrac{1 + n}{1 + |\Gamma_2 u_k| * |\Gamma_2 u_f|}, \text{при } |\Gamma_2 u_k| * |\Gamma_2 u_f| > 1 \\ 1 + \dfrac{n + |\Gamma_2 u_k| * |\Gamma_2 u_f|}{2}, \text{иначе} \end{cases}.$$

Следовательно:

- в лучшем случае при $|\Gamma_2 u_k| * |\Gamma_2 u_f| = 1$: $K_{UnionEdges}^{max} = 1 + \frac{1+n}{2}$;
- в худшем случае при $|\Gamma_2 u_k| * |\Gamma_2 u_f| = 0, n = 1$: $K_{UnionEdges}^{min} = 1.5$.

2.7. Декомпозиция вершины графа

Для операции входными данными являются:

1) x_k – разбиваемая вершина;
2) x_r, x_t – новые вершины, полученные в результате декомпозиции вершины x_k;
3) $U^{r'}$ – множество входящих ребер, которым инцидентна вершина x_r, $0 \le \left|U^{r'}\right| \le m$;
4) $U^{r''}$ – множество исходящих ребер, которые инцидентны вершине x_r, $0 \le \left|U^{r''}\right| \le$ $\le m$;
5) $U^{t\prime}$ – множество входящих ребер, которым инцидентна вершина x_t, $0 \le \left|U^{t'}\right| \le m$;
6) $U^{t\prime\prime}$ – множество исходящих ребер, которые инцидентны вершине x_t, $0 \le \left|U^{t''}\right| \le$ $\le m$;
7) u_f – вводимое ребро;
8) $X^{f'}$ – множество входящих вершин, которые инцидентны ребру u_f, $0 \le \left|X^{f'}\right| \le n$;
9) $X^{f''}$– множество исходящих вершин, которым инцидентно ребро u_f, $0 \le \left|X^{f''}\right| \le$ $\le n$.

Реализация операции предполагает:

- удаление вершины x_k из графа $G(X, U, \Gamma_1 X, \Gamma_2 U)$:

$$P_{DeleteVertex}(G, x_k);$$

- добавление вершины x_t в граф $G(X, U, \Gamma_1 X, \Gamma_2 U)$:

$$P_{AddVertex}\left(G, x_t, U^{t'}, U^{t''}\right);$$

- добавление ребра u_f в граф $G(X, U, \Gamma_1 X, \Gamma_2 U)$:

$$P_{AddEdge}\left(G, u_f, X^{f'}, X^{f''}\right).$$

Параллельную реализацию операции над графом представим последовательностью параллельно реализованных операций над графом:

$$P_{SplitVertex}\left(G, x_k, x_r, U^{r\prime}, U^{r\prime\prime}, x_t, U^{t\prime}, U^{t\prime\prime}, u_f, X^{f\prime}, X^{f''}\right) =>$$
$$=> [P_{DeleteVertex}(G, x_k), P_{AddVertex}\left(G, x_r, U^{r'}, U^{r''}\right), P_{AddVertex}\left(G, x_t, U^{t'}, U^{t''}\right),$$
$$P_{AddEdge}\left(G, u_f, X^{f'}, X^{f''}\right).$$

Рассчитаем коэффициент ускорения при использовании параллельной реализации операции.

Времена выполнения параллельной и последовательной операций складывается из времен операций, из которых они состоят соответственно. При $t(Op_{AddEl}) \cong t(Op_{DeleteEl})$:

- время выполнения параллельной операции:

$$t\left(P_{SplitVertex}\right) = t(P_{DeleteVertex}) + t(P_{AddVertex}) + t(P_{AddVertex}) + t(P_{AddEdge}) =$$
$$= t(Op_{DeleteEl}) + \left(1 + \left|U^{r''}\right|\right) * t(Op_{AddEl}) + \left(1 + \left|U^{t''}\right|\right) * t(Op_{AddEl}) + \left(1 + \left|X^{f''}\right|\right) *$$
$$* t(Op_{AddEl}) = \left(4 + \left|U^{r''}\right| + \left|U^{t''}\right| + \left|X^{f''}\right|\right) * t(Op_{AddEl});$$

- время выполнения последовательной операции:

$$t\left(\overline{P_{SplitVertex}}\right) = t\left(\overline{P_{DeleteVertex}}\right) + t\left(\overline{P_{AddVertex}}\right) + t\left(\overline{P_{AddVertex}}\right) + t\left(\overline{P_{AddEdge}}\right) =$$
$$= (2 + m) * t(Op_{DeleteEl}) + \left(2 + \left|U^{r'}\right| + \left|U^{r''}\right|\right) * t(Op_{AddEl}) + \left(2 + \left|U^{t'}\right| + \left|U^{t''}\right|\right) *$$
$$* t(Op_{AddEl}) + \left(2 + \left|X^{f'}\right| + \left|X^{f''}\right|\right) * t(Op_{AddEl}) =$$

$$= \left(8 + m + \left|U^{r'}\right| + \left|U^{r''}\right| + \left|U^{t'}\right| + \left|U^{t''}\right| + \left|X^{f'}\right| + \left|X^{f''}\right|\right) * t(Op_{AddEl}).$$

Откуда коэффициент ускорения:

$$K_{SplitVertex} = \frac{t\left(\overline{P_{SplitVertex}}\right)}{t\left(P_{SplitVertex}\right)} =$$

$$= \frac{\left(8 + m + \left|U^{r'}\right| + \left|U^{r''}\right| + \left|U^{t'}\right| + \left|U^{t''}\right| + \left|X^{f'}\right| + \left|X^{f''}\right|\right) * t(Op_{AddEl})}{\left(4 + \left|U^{r''}\right| + \left|U^{t''}\right| + \left|X^{f''}\right|\right) * t(Op_{AddEl})} =$$

$$= 1 + \frac{4 + m + \left|U^{r'}\right| + \left|U^{t'}\right| + \left|X^{f'}\right|}{4 + \left|U^{r''}\right| + \left|U^{t''}\right| + \left|X^{f''}\right|}.$$

Следовательно:

- в лучшем случае при $\left|U^{r'}\right| = \left|U^{t'}\right| = m, \left|X^{f'}\right| = n$ и $\left|U^{r''}\right| = \left|U^{t''}\right| = \left|X^{f''}\right| =$ $= 0$: $K_{UnionEdges}^{max} = 2 + \frac{3m+n}{4}$;
- в худшем случае при $\left|U^{r'}\right| = \left|U^{t'}\right| = \left|X^{f'}\right| = 0$ и $\left|U^{r''}\right| = \left|U^{t''}\right| = m, \left|X^{f''}\right| =$ $= n$: $K_{UnionEdges}^{min} = 1 + \frac{4+m}{4+2m+n}$.

2.8. Подразбиение ребра графа

С точки зрения реализации операция аналогична п. 2.7. Для данной операции входными данными являются:

1) u_k – разбиваемое ребро;
2) u_r, u_t – новые ребра;
3) $X^{r\prime}$ – множество вершин, которые инцидентны ребру u_r, $0 \le \left|X^{r'}\right| \le n$;
4) $X^{r\prime\prime}$ – множество вершин, которым инцидентно ребро u_r, $0 \le \left|X^{r''}\right| \le n$.
5) $X^{t'}$ – множество вершин, которые инцидентны ребру u_t, $0 \le \left|X^{t'}\right| \le n$;
6) $X^{t\prime\prime}$ – множество вершин, которым инцидентно ребро U_t, $0 \le \left|X^{t''}\right| \le n$;
7) x_f – вводимая вершина;
8) $U^{f'}$ – множество входящих ребер, которым инцидентна вершина X_f, $0 \le \left|U^{f'}\right| \le$ $\le m$;
9) $U^{f''}$– множество исходящих ребер, которые инцидентны вершине X_f, $0 \le \left|U^{f''}\right| \le$ $\le m$;

Параллельную реализацию операции над графом представим последовательностью параллельно реализованных операций над графом:

$$P_{SplitEdge}\left(G, u_k, u_r, X^{r\prime}, X^{r\prime\prime}, u_t, X^{t\prime}, X^{t\prime\prime}, x_f, U^{f\prime}, U^{f''}\right) =>$$

$$=> [P_{DeleteEdge}(G, u_k), P_{AddEdge}\left(G, u_r, X^{r'}, X^{r''}\right), P_{AddEdge}\left(G, u_t, X^{t'}, X^{t''}\right),$$

$$P_{AddVertex}\left(G, x_f, U^{f'}, U^{f''}\right).$$

Рассчитаем коэффициент ускорения при использовании параллельной реализации операции.

При $t(Op_{AddEl}) \cong t(Op_{DeleteEl})$:

- время выполнения параллельной операции:

$$t(P_{SplitEdge}) = t(P_{DeleteEdge}) + t(P_{AddEdge}) + t(P_{AddEdge}) + t(P_{AddVertex}) =$$
$$= t(Op_{DeleteEl}) + (1 + |X^{r''}|) * t(Op_{AddEl}) + (1 + |X^{t''}|) * t(Op_{AddEl}) + (1 + |U^{f''}|) *$$
$$* t(Op_{AddEl}) = (4 + |X^{r''}| + |X^{t''}| + |U^{f''}|) * t(Op_{AddEl});$$

- время выполнения последовательной операции:

$$t(\overline{P_{SplitEdge}}) = t(\overline{P_{DeleteEdge}}) + t(\overline{P_{AddEdge}}) + t(\overline{P_{AddEdge}}) + t(\overline{P_{AddVertex}}) =$$
$$= (2 + n) * t(Op_{DeleteEl}) + (2 + |X^{r'}| + |X^{r''}|) * t(Op_{AddEl}) + (2 + |X^{t'}| + |X^{t''}|) *$$
$$* t(Op_{AddEl}) + (2 + |U^{f'}| + |U^{f''}|) * t(Op_{AddEl}) =$$
$$= (8 + n + |X^{r'}| + |X^{r''}| + |X^{t'}| + |X^{t''}| + |U^{f'}| + |U^{f''}|) * t(Op_{AddEl}).$$

Откуда коэффициент ускорения:

$$K_{SplitEdge} = \frac{t(\overline{P_{SplitEdge}})}{t(P_{SplitEdge})} =$$
$$= \frac{(8 + n + |X^{r'}| + |X^{r''}| + |X^{t'}| + |X^{t''}| + |U^{f'}| + |U^{f''}|) * t(Op_{AddEl})}{(4 + |X^{r''}| + |X^{t''}| + |X^{f''}|) * t(Op_{AddEl})} =$$
$$= 1 + \frac{4 + n + |X^{r'}| + |X^{t'}| + |U^{f'}|}{4 + |X^{r''}| + |X^{t''}| + |U^{f''}|}.$$

Следовательно:

- в лучшем случае при $|X^{r'}| = |X^{t'}| = n, |U^{f'}| = m$ и $|X^{r''}| = |X^{t''}| = |U^{f''}| =$ $= 0$: $K_{SplitEdge}^{max} = 2 + \frac{3n+m}{4}$;
- в худшем случае при $|X^{r'}| = |X^{t'}| = |U^{f'}| = 0$ и $|X^{r''}| = |X^{t''}| = n, |U^{f''}| =$ $= m$: $K_{SplitEdge}^{min} = 1 + \frac{4+n}{4+2n+m}$.

2.9. Удаление вершины из образов и прообразов инцидентных ей ребер

Для операции входными данными являются:

1) x_k – удаляемая из образов и прообразов вершина;
2) U' – множество входящих ребер, которым инцидентна вершина $x_k, 0 \leq |U'| \leq m$;
3) U'' – множество исходящих ребер, которые инцидентны вершине $x_k, 0 \leq |U''| \leq$ $\leq m$.

Реализация операции предполагает:

- удаление ребер U' из образа $\Gamma_1 x_k$:

$$\Gamma_1 x_k \backslash U';$$

- множество операций удаления вершины x_k из образов инцидентных ей ребер:

$$\{\Gamma_2 U_i \backslash x_k, i = \overline{1, m}\}.$$

Параллельная операция содержит уже введенную элементарную операцию удаления элемента из множества (2).

Тогда реализация параллельной операции принимает вид:

$$P_{DeleteVertexFromEdges}(G, x_k, U', U'') =>$$

$$=> \left[\begin{array}{l} \{Op_{DeleteEl}(\Gamma_1 x_k, U'_i), i = \overline{1,p}, p = |U'|\}, \\ \{Op_{DeleteEl}(\Gamma_2 U''_1, x_k), \\ \dots \\ Op_{DeleteEl}(\Gamma_2 U''_q, x_k), q = |U''|\}. \end{array}\right.$$

Рассчитаем коэффициент ускорения при использовании параллельной реализации операции:

- время выполнения параллельной операции:

$$t(P_{DeleteVertexFromEdges}) = \begin{cases} t(|U'| * Op_{DeleteEl}), \text{при } |U'| > 1 \\ t(Op_{DeleteEl}), \text{иначе} \end{cases} =$$

$$= \begin{cases} |U'| * t(Op_{DeleteEl}), \text{при } |U'| > 1 \\ t(Op_{DeleteEl}), \text{иначе} \end{cases};$$

- время выполнения последовательной операции:

$$t(\overline{P_{DeleteVertexFromEdges}}) = t((|U'| + |U''|) * Op_{DeleteEl}) = (|U'| + |U''|) * t(Op_{DeleteEl}).$$

Откуда коэффициент ускорения:

$$K_{DeleteVertexFromEdges} = \frac{t(\overline{P_{DeleteVertexFromEdges}})}{t(P_{DeleteVertexFromEdges})} =$$

$$= \begin{cases} \dfrac{(|U'| + |U''|) * t(Op_{DeleteEl})}{|U'| * t(Op_{DeleteEl})}, \text{при } |U'| > 1 \\ \dfrac{(|U'| + |U''|) * t(Op_{DeleteEl})}{t(Op_{DeleteEl})}, \text{иначе} \end{cases} = \begin{cases} \dfrac{(|U'| + |U''|)}{|U'|}, \text{при } |U'| > 1 \\ |U'| + |U''|, \text{иначе} \end{cases}.$$

Следовательно:

- в лучшем случае при $|U'| = 1, |U''| = m$: $K^{max}_{DeleteVertexFromEdges} = 1 + m$;
- в худшем случае при $|U'| = m, |U''| = 0$: $K^{min}_{DeleteVertexFromEdges} = 1$.

2.10. Удаление ребра из образов и прообразов инцидентных ему вершин

С точки зрения реализации операция аналогична операции, описанной в п. 2.9. Входными данными являются:

1) u_k – удаляемая из образов и прообразов вершина;
2) X' – множество вершин, которые инцидентны ребру $u_k, 0 \leq |X'| \leq n$;
3) X'' – множество вершин, которым инцидентно ребро $u_k, 0 \leq |X''| \leq n$.

Тогда реализация параллельной операции принимает вид:

$$P_{DeleteEdgeFromVertices}(G, u_k, X', X'') =>$$

$$=> \begin{cases} \{Op_{DeleteEl}(\{\Gamma_2 u_k, X_i'\}), i = \overline{1,p}, p = |X'|\}, \\ \{Op_{DeleteEl}(\Gamma_1 X_1'', u_k), \\ \dots \\ Op_{DeleteEl}(\Gamma_1 X_q'', u_k), q = |X''|\}. \end{cases}$$

Рассчитаем коэффициент ускорения при использовании параллельной реализации операции:

- время выполнения параллельной операции:

$$t(P_{DeleteEdgeFromVertices}) = \begin{cases} t(|X'| * Op_{DeleteEl}), \text{при } |X'| > 1 \\ t(Op_{DeleteEl}), \text{иначе} \end{cases} =$$

$$= \begin{cases} |X'| * t(Op_{DeleteEl}), \text{при } |X'| > 1 \\ t(Op_{DeleteEl}), \text{иначе} \end{cases};$$

- время выполнения последовательной операции:

$$t(\overline{P_{DeleteEdgeFromVertices}}) = t((|X'| + |X''|) * Op_{DeleteEl}) = (|X'| + |X''|) * t(Op_{DeleteEl})$$

Откуда коэффициент ускорения:

$$K_{DeleteEdgeFromVertices} = \frac{t(\overline{P_{DeleteEdgeFromVertices}})}{t(P_{DeleteEdgeFromVertices})} =$$

$$= \begin{cases} \dfrac{(|X'| + |X''|) * t(Op_{DeleteEl})}{|X'| * t(Op_{DeleteEl})}, \text{при } |X'| > 1 \\ \dfrac{(|X'| + |X''|) * t(Op_{DeleteEl})}{t(Op_{DeleteEl})}, \text{иначе} \end{cases} = \begin{cases} \dfrac{(|X'| + |X''|)}{|X'|}, \text{при } |X'| > 1 \\ |X'| + |X''|, \text{иначе} \end{cases}.$$

Следовательно:

- в лучшем случае при $|X'| = 1, |X''| = n$: $K_{DeleteEdgeFromVertices}^{max} = 1 + n$;
- в худшем случае при $|X'| = n, |X''| = 0$: $K_{DeleteEdgeFromVertices}^{min} = 1$.

Заключение

В результате анализа было выявлено множество элементарных операций, представленные выражениями (1) – (5), описаны параллельные реализации операций и выполнена в соответствии с принятыми допущениями оценка «снизу» коэффициента ускорения основных операций над графами при применении параллельных алгоритмов. Результаты анализа сведены в таблицу 1.

Таблица 1. Результаты анализа операций

Хар-ки / Операции	Вхождение элементарной операции					Оценка ускорения	
	Op_{AddEl}	Op_{AddSet}	$Op_{DeleteEl}$	$Op_{DeleteSet}$	$Op_{UnionSet}$	K^{min}	K^{max}
1) Добавление вершины	+	+				$\frac{2+m}{1+m}$	$2+m$
2) Добавление ребра	+	+				$\frac{2+n}{1+n}$	$2+n$
3) Удаление вершины			+	+		$2+m$	$2+m$
4) Удаление ребра			+	+		$2+n$	$2+n$
5) Свертка вершин			+	+	+	1.5	$1+\frac{1+m}{2}$
6) Стягивание ребер			+	+	+	1.5	$1+\frac{1+n}{2}$
7) Декомпозиция вершин	+	+	+	+		$1+\frac{4+m}{4+2m+n}$	$2+\frac{3m+n}{4}$
8) Подразбиение ребер	+	+	+	+		$1+\frac{4+n}{4+2n+m}$	$2+\frac{3n+m}{4}$
9) Удаление вершины из образов и прообразов			+			1	$1+m$
10) Удаление ребра из образов и прообразов			+			1	$1+n$

Среди рассмотренных операций минимальное значение коэффициента ускорения равно единице для операций удаления вершины и ребра из образов и прообразов. Максимальный коэффициент ускорения больше либо равен единице и прямо пропорционален размерностям графа. Так, например, для операции декомпозиции вершин для графа с количеством вершин $n = 10^6$ и количеством ребер $m = 10^4$ максимальное значение коэффициента ускорения $K^{max} = 2.575 * 10^5$, минимальное - $K^{min} = 1.083$. Приведенные оценки коэффициента ускорения показывают высокую эффективность применения параллельных алгоритмов представленных операций.

Перспективными направлениями дальнейших исследований в области ускорения обработки графовой модели является анализ операций преобразования, таких как пересечение, объединение, композиция графов и т.д., и операций анализа характеристик графа, а также исследования в области построения структур данных, обеспечивающих максимальную степень распараллеливания обработки, что позволит увеличить эффективность использования параллельных вычислительных систем для обработки графов.

Список литературы

1. Овчинников В.А., Иванова Г.С., Ничушкина Т.Н. Выбор структур данных для представления графов при решении комбинаторно-оптимизационных задач // Вестник МГТУ им. Н.Э. Баумана. Сер. Приборостроение. 2001. № 2 (43). С. 39-51.
2. Иванова Г.С. Методология и средства разработки алгоритмов решения задач анализа и синтеза структур программного обеспечения и устройств вычислительной техники: дис. … докт. техн. наук. М., 2007. 416 с.
3. Овчинников В.А. Операции над ультра и гиперграфами для реализации процедур анализа и синтеза структур сложных систем // Наука и образование. МГТУ им. Н.Э. Баумана. Электрон. журн. 2009. № 10. Режим доступа: http://technomag.bmstu.ru/doc/132769.html (дата обращения 20.10.2014).
4. Овчинников В.А. Операции над ультра и гиперграфами для реализации процедур анализа и синтеза структур сложных систем (часть 2) // Наука и образование. МГТУ им. Н.Э. Баумана. Электрон. журн. 2009. № 11. Режим доступа: http://technomag.bmstu.ru/doc/133223.html (дата обращения 20.10.2014).
5. Овчинников В.А. Операции над ультра и гиперграфами для реализации процедур анализа и синтеза структур сложных систем (часть 3) // Наука и образование. МГТУ им. Н.Э. Баумана. Электрон. журн. 2009. № 12. Режим доступа: http://technomag.bmstu.ru/doc/134335.html (дата обращения 20.10.2014).
6. Головков А.А. Представление графовых моделей в системах параллельной обработки // Молодежный научно-технический вестник. МГТУ им. Н.Э. Баумана. Электрон. журн. 2014. № 7. Режим доступа: http://sntbul.bmstu.ru/doc/727773.html (дата обращения 20.10.2014).
7. Чернышев С.В. Модели, методы и алгоритмы эффективного решения задачи маршрутизации транспорта на графах больших размерностей: дис. … канд. техн. наук. М., 2011. 116 с.
8. Leskovec J., Huttenlocher D., Kleinberg J. Predicting Positive and Negative Links in Online Social Networks // Proceedings of the 19th International Conference on World Wide Web (WWW’10). ACM, 2010. P. 641-650. DOI: 10.1145/1772690.1772756
9. Bullmore E., Sporns O. Complex brain networks: graph theoretical analysis of structural and functional systems // Nature Reviews Neuroscience. 2009. Vol. 10, no. 3. P. 186-198. DOI: 10.1038/nrn2575.
10. deLorimier M. GRAph Parallel Actor Language — A Programming Language for Parallel Graph Algorithms. PhD Thesis. California Institute of Technology, Pasadena, California,

2013. 151 p. Available at:
http://thesis.library.caltech.edu/7188/2/delorimier_Michael_2013.pdf, accessed 20.10.2014.

11. Xin R.S., Crankshaw D., Crankshaw D., Dave A., Gonzalez J.E., Franklin M.J., Stoica I. GraphX: Unifying Data-Parallel and Graph-Parallel Analytics. UC Berkeley AMPLab, 2014. 5 p. Available at: http://www.researchgate.net/publication/260147249_GraphX_Unifying_Data-Parallel_and_Graph-Parallel_Analytics, accessed 20.10.2014.
12. Harish Pawan, Narayanan P.J. Accelerating Large Graph Algorithms on the GPU Using CUDA // In: High Performance Computing –Proceedings of the 14th international conference on High performance computing (HiPC'07) / S. Aluru, et al. (eds.). Springer Berlin Heidelberg, 2007. P. 197-208. (Ser. Lecture Notes in Computer Science; vol. 4873). DOI: 10.1007/978-3-540-77220-0_21
13. Merrill D., Garland M., Grimshaw A. Scalable GPU Graph Traversal // Proceedings of the 17th ACM SIGPLAN symposium on Principles and Practice of Parallel Programming (PPoPP'12). ACM, 2012. P. 117-128. DOI: 10.1145/2145816.2145832

Science&Education of the Bauman MSTU
Electronic journal
ISSN 1994-0448

Science and Education of the Bauman MSTU, 2014, no. 11, pp. 535–554.

DOI: **10.7463/1114.0741563**

Received: 23.11.2014
Revised: 29.11.2014

Evaluating Efficiency of Parallel Algorithms of Transformation Operations with Graph Model

G.S. Ivanova[1,*], **A.A. Golovkov**[1]

[*]gsivanova@gmail.com

[1]Bauman Moscow State Technical University, Moscow, Russia

Keywords: graphs, operations with graphs, parallel algorithms, parallel computing, speedup

The usage of graphs in the analysis and design of complex large-scale system structures, which require a significant computing capacity, has led to the need to seek for the new methods both of graph models representation and of graph operations implementation. To reduce the execution time of algorithms the parallel computing systems are appropriate to use. In this case, to achieve the maximum acceleration graph processing is implemented in hardware and software parts of the system.

In the article, analysis of graph model operations was performed in the context of their implementation in a parallel computing system based on the abstract description of the graph by the sets that allows the utilizing of various parallel processing systems, regardless of their architecture features. The most common in the algorithms graph transformation operations were considered.

As a result of analysis a set of elementary operations on graph structures, which constitute graph operations, was revealed, and graph operations parallel algorithms were realized. Efficiency evaluation of parallel algorithms, presented by speedup for each operation, showed a high degree of graph processing acceleration, compared with sequential operations algorithms.

The proposed realization can be used to solve time-consuming large-scale tasks on parallel computing systems. At the same time, in a particular parallel processing system it is also possible to parallelize elementary operations thereby greatly reducing the execution time of operation in general.

Further research is focused on describing and complementing current implementation with complex parallel graph transformation operations such as intersection, union, composition of graphs, etc., as well as analysis operations of various graph characteristics. That will expand the set of elementary operations and will provide an opportunity to evaluate more accurately the efficiency of parallel computing systems for processing graph models.

References

1. Ovchinnikov V.A., Ivanova G.S., Nichushkina T.N. Selection of Data Structures for Graph Representation while Solving Combinatorial and Optimizational Problems. *Vestnik MGTU. Ser. Priborostroenie = Herald of the Bauman MSTU. Ser. Instrument Engineering*, 2001, no. 2, pp. 39-51. (in Russian).
2. Ivanova G.S. *Metodologiya i sredstva razrabotki algoritmov resheniya zadach analiza i sinteza struktur programmnogo obespecheniya i ustroystv vychislitel'noy tekhniki. Dokt. diss.* [Methodology and tools for development algorithms for solving problems of analysis and synthesis of structures of software and computing devices. Dr. dis.]. Moscow, 2007. 416 p. (in Russian).
3. Ovchinnikov V.A. Operations over ultra- and hypercolumns for realisation of procedures of the analysis and synthesis of structures of difficult systems. *Nauka i obrazovanie MGTU im. N.E. Baumana = Science and Education of the Bauman MSTU*, 2009, no. 10. Available at: http://technomag.bmstu.ru/doc/132769.html , accessed 20.10.2014. (in Russian).
4. Ovchinnikov V.A. Operations over ultra- and hypercolumns for realisation of procedures of the analysis and synthesis of structures of difficult systems (Part 2). *Nauka i obrazovanie MGTU im. N.E. Baumana = Science and Education of the Bauman MSTU*, 2009, no. 11. Available at: http://technomag.bmstu.ru/doc/133223.html , accessed 20.10.2014. (in Russian).
5. Ovchinnikov V.A. Operations over ultra- and hypercolumns for realisation of procedures of the analysis and synthesis of structures of difficult systems (Part 3). *Nauka i obrazovanie MGTU im. N.E. Baumana = Science and Education of the Bauman MSTU*, 2009, no. 12. Available at: http://technomag.bmstu.ru/doc/134335.html , accessed 20.10.2014. (in Russian).
6. Golovkov A.A. Representation of graph models in parallel processing systems. *Molodezhnyi nauchno-tekhnicheskii vestnik MGTU im. N.E. Baumana = Youth Science and Technology Herald of the Bauman MSTU*, 2014, no. 7. Available at: http://sntbul.bmstu.ru/doc/727773.html , accessed 20.10.2014. (in Russian).
7. Chernyshev S.V. *Modeli, metody i algoritmy effektivnogo resheniya zadachi marshrutizatsii transporta na grafakh bol'shikh razmernostey. Kand. dis*. [Models, methods and algorithms for efficient solving the vehicle routing problem on large-scale graphs. Cand. dis.]. Moscow, 2011. 116 p. (in Russian).
8. Leskovec J., Huttenlocher D., Kleinberg J. Predicting Positive and Negative Links in Online Social Networks. *Proceedings of the 19th International Conference on World Wide Web (WWW'10)*. ACM, 2010, pp. 641-650. DOI: 10.1145/1772690.1772756
9. Bullmore E., Sporns O. Complex brain networks: graph theoretical analysis of structural and functional systems. *Nature Reviews Neuroscience*, 2009, vol. 10, no. 3, pp. 186-198. DOI: 10.1038/nrn2575.

10. deLorimier M. *GRAph Parallel Actor Language — A Programming Language for Parallel Graph Algorithms. PhD Thesis*. California Institute of Technology, Pasadena, California, 2013. 151 p. Available at: http://thesis.library.caltech.edu/7188/2/delorimier_Michael_2013.pdf, accessed 20.10.2014.
11. Xin R.S., Crankshaw D., Crankshaw D., Dave A., Gonzalez J.E., Franklin M.J., Stoica I. *GraphX: Unifying Data-Parallel and Graph-Parallel Analytics*. UC Berkeley AMPLab, 2014. 5 p. Available at: http://www.researchgate.net/publication/260147249_GraphX_Unifying_Data-Parallel_and_Graph-Parallel_Analytics, accessed 20.10.2014.
12. Harish Pawan, Narayanan P.J. Accelerating Large Graph Algorithms on the GPU Using CUDA. In: Aluru S., et al., eds. *High Performance Computing –Proceedings of the 14th international conference on High performance computing (HiPC'07)*. Springer Berlin Heidelberg, 2007, pp. 197-208. (Ser. *Lecture Notes in Computer Science*; vol. 4873). DOI: 10.1007/978-3-540-77220-0_21
13. Merrill D., Garland M., Grimshaw A. Scalable GPU Graph Traversal. *Proceedings of the 17th ACM SIGPLAN symposium on Principles and Practice of Parallel Programming (PPoPP'12).* ACM, 2012, pp. 117-128. DOI: 10.1145/2145816.2145832