Реализация индексного анализа для деревьев циклов любого вида сложности

Выполнил: студент 818 гр. Юдин Павел

Научный руководитель: к.т.н. Муханов Л.Е.

Актуальность

1. Увеличение производительности вычислительных комплексов. Многоядерные архитектуры
2. Значительное количество существующих приложений реализовано для последовательного исполнения
3. Автоматическое распараллеливание позволяет повышать производительность приложений без изменений в исходном коде

Актуальность

- В большинстве вычислительных задач основное время тратится на вычисления, которые содержатся внутри циклов
- Автоматическое распараллеливание нацелено на работу с циклами

Основные понятия

- Индексный анализ – анализ, проводящийся над индексами массивов в определенном цикле, для выявления зависимостей между конкретными операциями на разных итерациях
- Дерево циклов – структура, выражающая отношения вложенности циклов
- Гнездо циклов – структура, содержащая информацию об одной ветви в дереве циклов

Распараллеливание циклов

- Межитерационная цикловая зависимость

```
for (i=0;i<10;i++)
{
  a[i+1]=a[i]+5;
}
```

→ существует зависимость

```
for(i=0;i<10;i++)
{
   a[i]=a[i]+5;
}
```

→ отсутствует зависимость

- Цикловой индексный анализ позволяет определить отсутствие зависимостей

Проблематика

- Невозможность анализировать операции, принадлежащие разным гнездам

```
for(i=0;i<1000;i++)
{
    for(k=0;k<1000;k++)
        a[i][k]+=1;
    for(j=0;j<1000;j++)
        a[i][j]+=1;
}
```

Постановка задачи

1. Разбор существующих методов анализа межитерационных цикловых зависимостей
2. Разбор программной реализации существующих методов
3. Реализация анализа для деревьев циклов любого вида сложности

Математическая постановка

$$
\begin{aligned}
&for(int\ i_1 = A_1; i_1 < B_1; i++)\{ \\
&\quad \dots \\
&\quad\quad for(int\ i_n = A_n; i_n < B_n; i_n++) \\
&\quad \dots \\
&\quad\quad\quad a[f_1(\vec{i})]...[f_n(\vec{i})] = ... \\
&\quad \dots \\
&\quad = a[h_1(\vec{i})]...[h_n(\vec{i})]; \\
&\}
\end{aligned}
$$

$$
\begin{aligned}
&I = \{i_1, \dots, i_n\} \\
&f_1(\vec{i}) = f_1(\xi_1) \qquad h_1(\vec{i}) = h_1(\alpha_1) \\
&f_2(\vec{i}) = f_2(\xi_2) \qquad h_2(\vec{i}) = h_2(\alpha_2) \\
&\dots \qquad\qquad \dots \\
&f_n(\vec{i}) = f_n(\xi_n) \qquad h_n(\vec{i}) = h_n(\alpha_n) \\
&\text{где } \xi_1, \dots, \xi_n, \alpha_1, \dots, \alpha_n \in I \\
&S_\xi = \{\xi_1, \dots, \xi_n\}, S_\alpha = \{\alpha_1, \dots, \alpha_n\} \\
&A = S_\xi \cap I, B = S_\alpha \cap I
\end{aligned}
$$

Исходная версия: Доказать независимость при эквивалентности множеств A и B

Новая версия: Доказать независимость вне зависимости от A == B

Представление индекса массива

Внутреннее представление

mov 10 => Vs_3

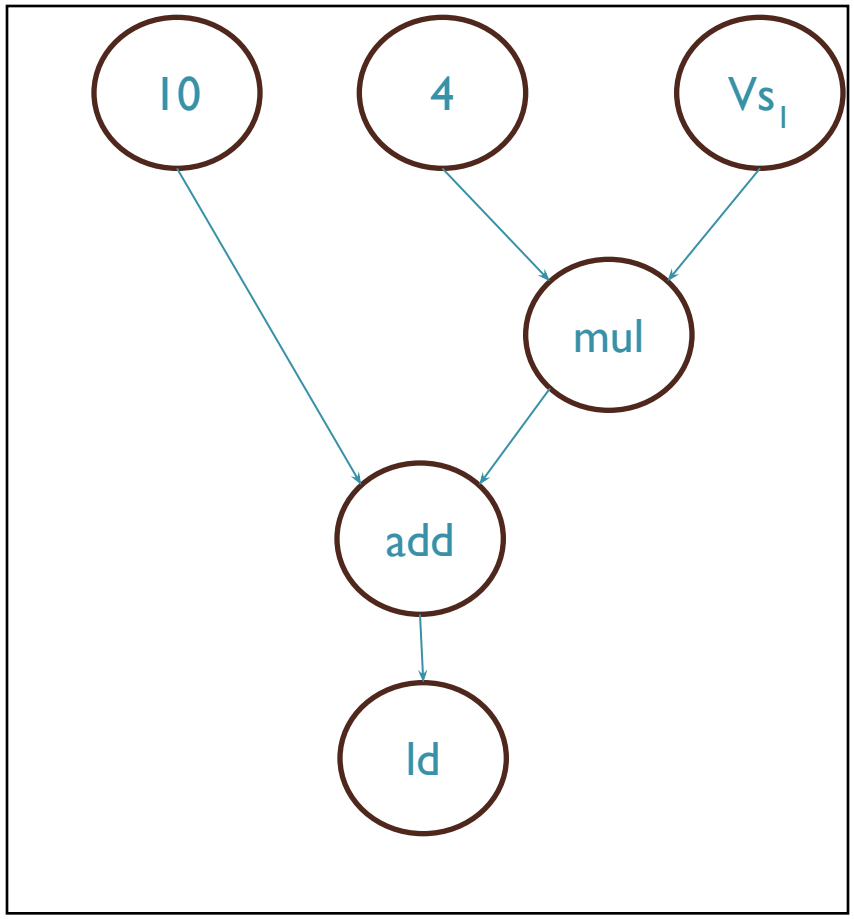
mov 4 => Vs_0

mul Vs_0,Vs_1 => Vs_2

add Vs_2,Vs_3 => Vs_4

load a[Vs_4]

Граф потока данных



PS- форма:

4*(Vs_1+10)

Формирование уравнений для анализа зависимостей

- Линейная форма представления PS-формы:

$c_0 + c_1 * x_1 + c_2 * x_2 + \ldots + c_n * x_n$

- Формирование неравенств:

```
for(i=0;i<10;i++)
{
    a[i]=a[i+10]+5;
}
```

→

$0 \le i < 10$

$0 \le j < 10$

$i = j + 10$

Методы решения

1. «одно ограничение на переменную»

-не охватывает все случаи

2. Ациклический

-не охватывает все случаи

3. Простая проверка остатка цикла

-не охватывает все случаи

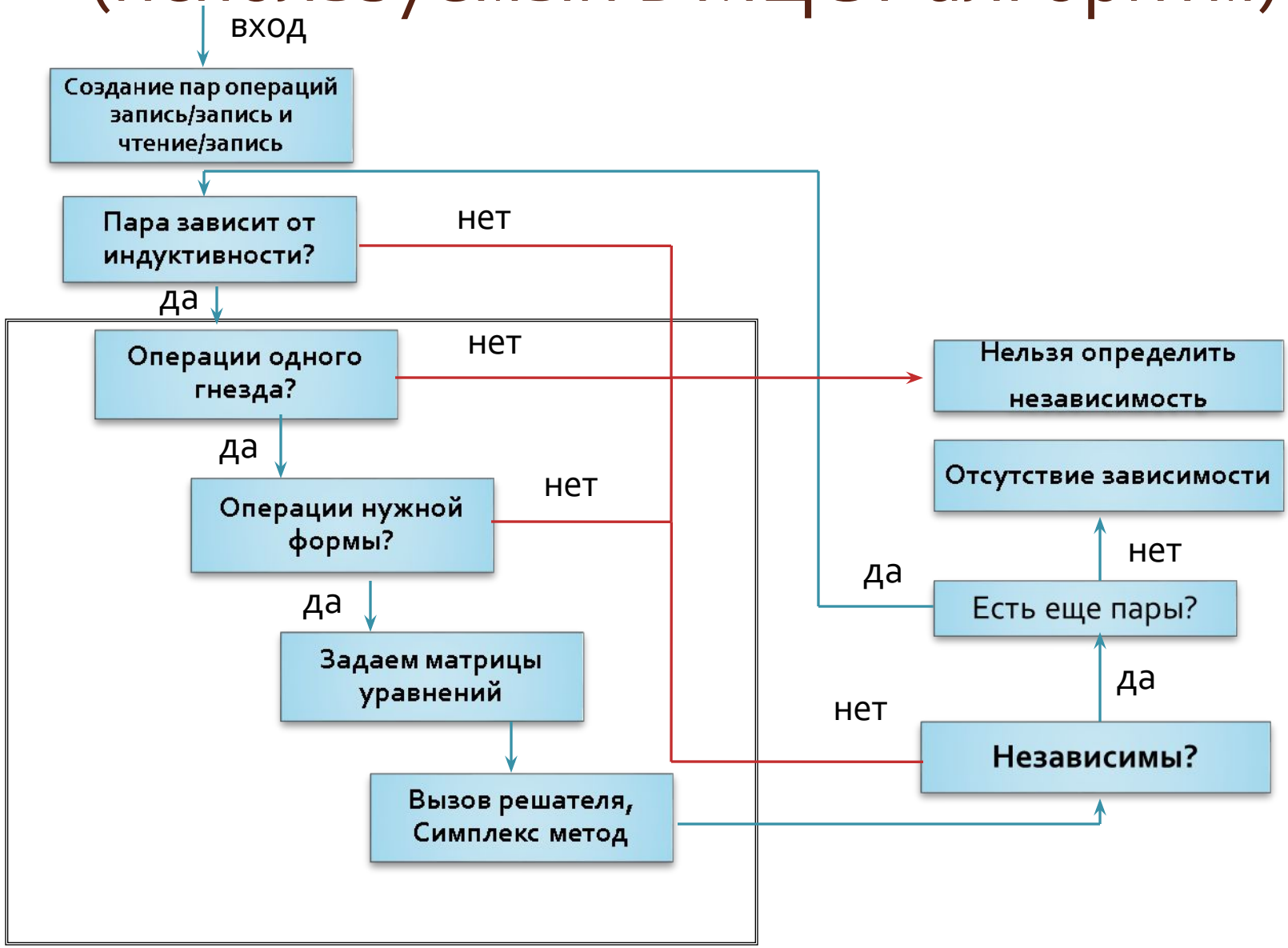
4. Фурье-Моцкина

-двойная экспоненциальная сложность

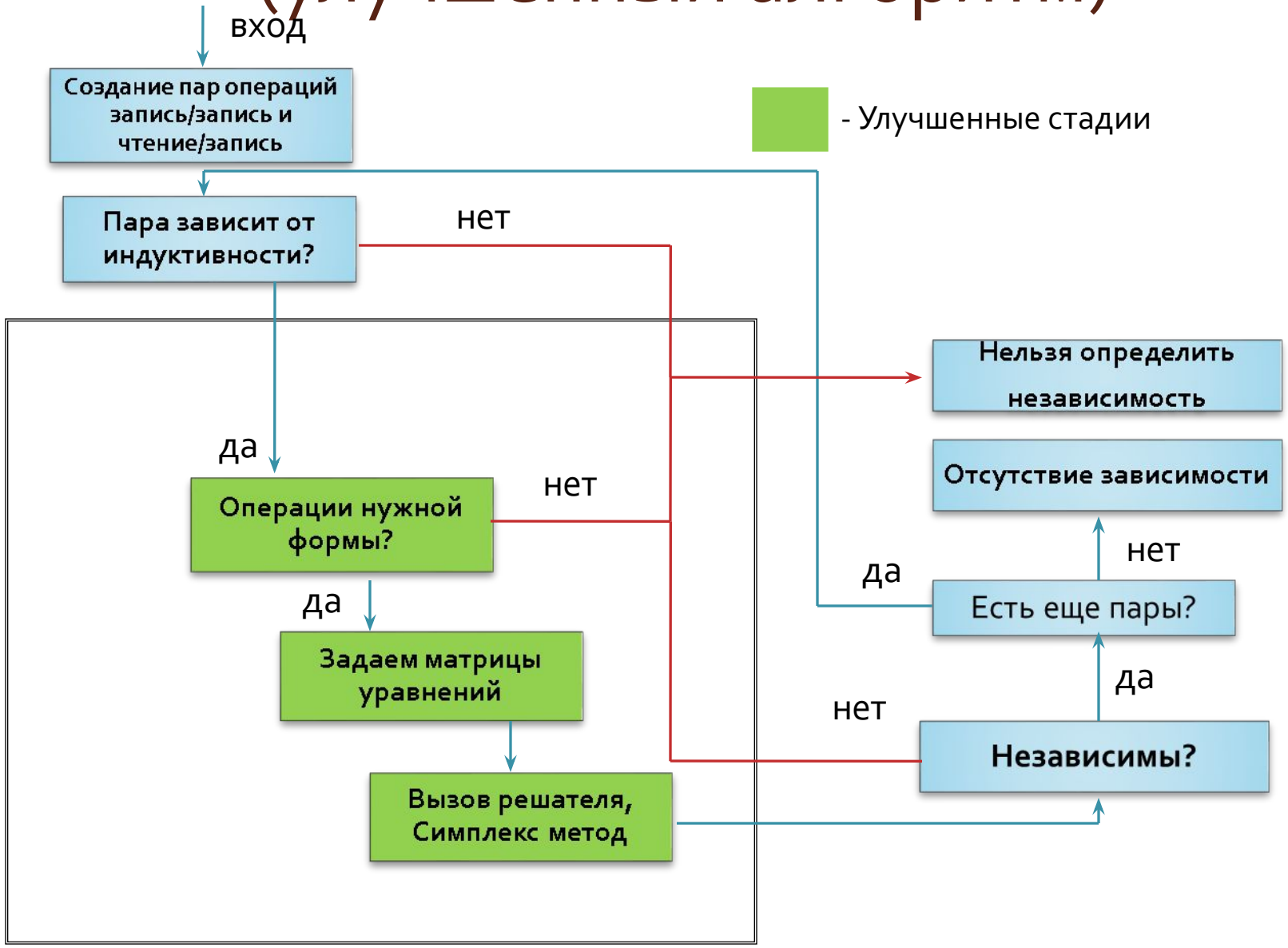
5. Симплекс метод

-охватывает все случаи, экспоненциальная сложность

Цикловой индексный анализ (используемый в МЦСТ алгоритм)



Цикловой индексный анализ (улучшенный алгоритм)

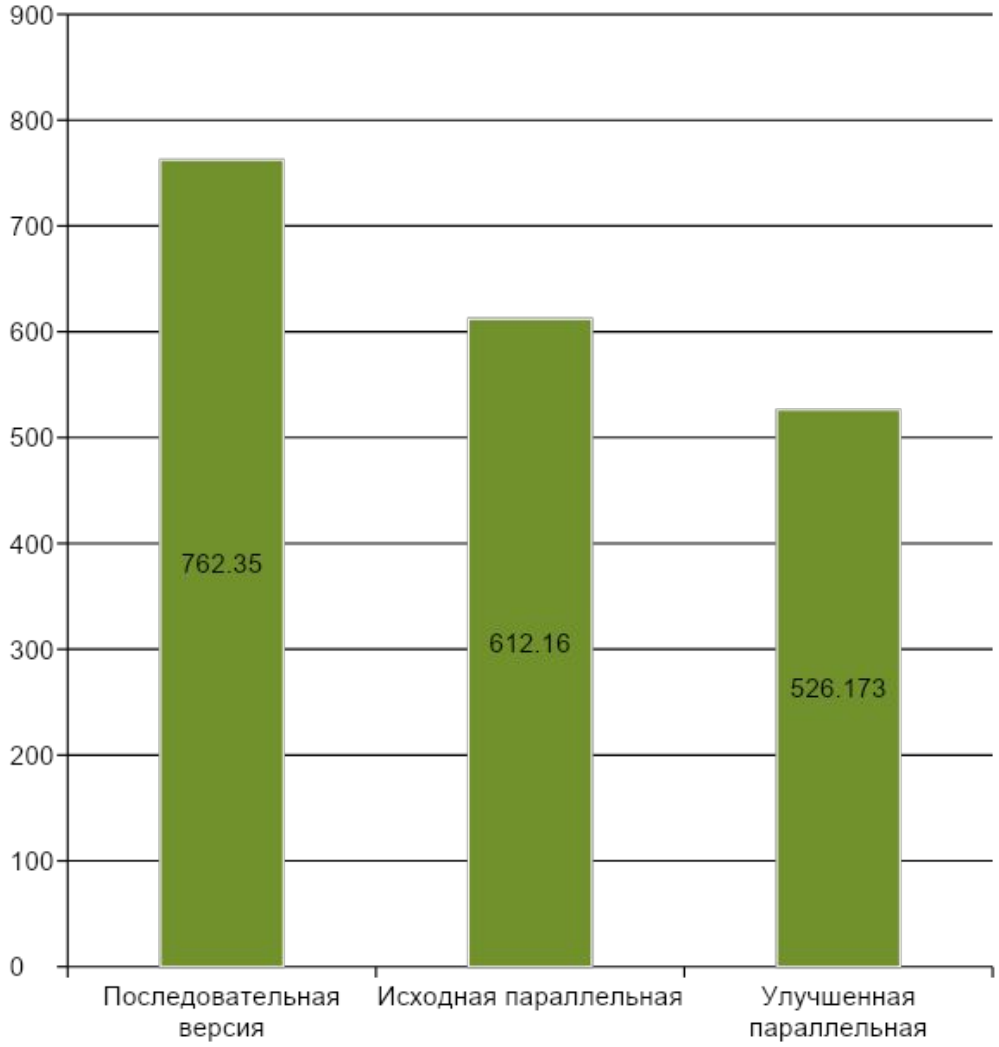


Экспериментальные результаты

Задача wupwise168

из пакета тестов Spec2000

Время параллельного выполнения задачи сократилось на 14%



Результаты

- ✔ Разобраны существующие методы анализа межитерационных цикловых зависимостей
- ✔ Разобрана программная реализация существующих методов
- ✔ Реализован анализ для деревьев циклов любого вида сложности
- ✔ Получены экспериментальные результаты на задаче **wupwise168**, пакета **Spec2000**
- ✔ Улучшения внедрены в компилятор