Новосибирский государственный университет
Факультет информационных технологий

ВКР бакалавра на тему:

Разработка и реализация алгоритмов конструирования частных управляющих схем для исполнения фрагментированных программ

Выполнил студент НГУ ФИТ
По специальности Информатика и ВТ
Группы 13201
Лысенко Егор Олегович
Научный руководитель:
к.т.н., доцент Маркова Валентина Петровна

Новосибирск, 2017 г.

Распределенные высокопроизводительные вычисления

Программы из области высокопроизводительных вычислений можно условно разбить на 2 части:

- **Прикладная часть** производит вычисления данных прикладной области.
- **Системная часть** управляет вычислениями для обеспечения производительности.

Реализация системной части требует специфических знаний. Объем системной части зависит от сложности параллельного вычислителя, от размера задачи.

Проблема

Абстрагирование от программиста системной части приводит к ее **неэффективности** на численных методах.

Актуальность

Абстрагирование системной части сокращает ее трудоемкость, сокращает объем необходимых знаний, позволяет решать задачи большего размера на численных методах.

Обзор средств программирования распределенных вычислений

	MPI и другие сетевые библиотеки	**DVM**	**charm++**	**LuNA**
Абстрагирование системной части на задачах численных методов	−	−+	+−	+
Производительность распределенных вычислений на задачах численных методов	+	+−	−+	−

Фрагментированное программирование. LuNA

Фрагментированное программирование - парадигма реализации крупномасштабных численных моделей, основанная на следующих **основных принципах**:

- Единственность записи данных.
- Разделение задач и данных.
- Исполнение задач по готовности аргументов (data-flow).
- Отсутствие побочных эффектов у задач.

Language for Numerical Algorithms (**LuNA**) реализует принципы фрагментированного программирования.
Возможно использовать существующие наработки как базу для вклада в решение поставленной проблемы. Для этого лучше всего подходит **LuNA.**

Цель работы

Цель данной работы - разработка и реализация алгоритмов автоматического конструирования частных управляющих схем, позволяющих снизить накладные расходы LuNA на задачах численных методов.

Требования

- Практическая значимость: существенное улучшение производительности LuNA на численных методах.
- Интегрированность в систему существующих средств оптимизаций LuNA.

Задачи

- Локализовать узкие места в алгоритмах LuNA.
- Разработать алгоритмы конструирования частных управляющих схем, снижающие накладные расходы LuNA в узких местах алгоритмов LuNA.
- Реализовать и протестировать часть разработанных алгоритмов конструирования частных управляющих схем, снижающих накладные расходы LuNA в узких местах алгоритмов LuNA.

Результаты работы

Аналитический обзор узких мест в алгоритмах LuNA

LuNA использует универсальные алгоритмы **распределения задач и данных на узлы** (**РЗДУ**), что на многих актуальных задачах становится причиной больших накладных расходов. **Это узкое место - первостепенное**, так как делает невозможным на многих актуальных задачах использование большого количества узлов.

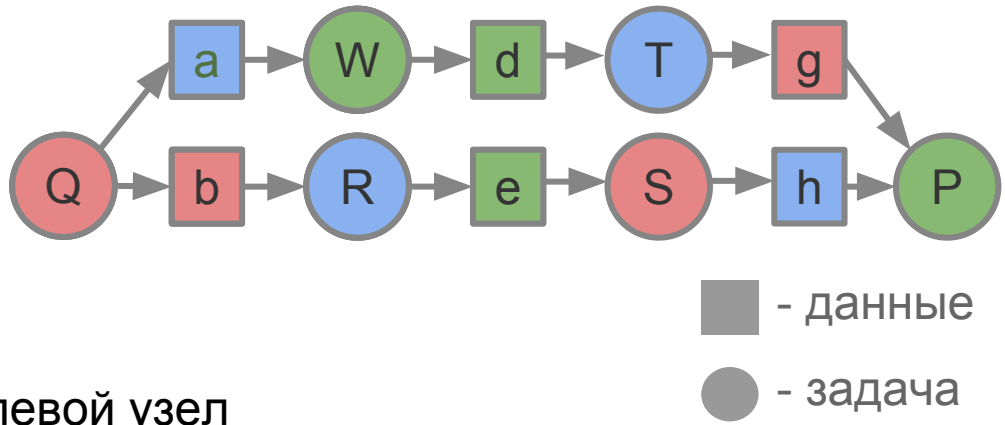
Также актуальны узкие места, связанные с:

- Порядком исполнения задач и степенью параллелизма.
- Накладными расходами в интерпретации LuNA языка.
- Хранением неиспользуемых данных.

Пример LuNA программы и ее РЗДУ на основе hash

```
1.  // fork
2.  a, b = Q()
3.  // первая ветка
4.  d = W(a)
5.  g = T(d)
6.  // вторая ветка
7.  e = R(b)
8.  h = S(e)
9.  // join
10. P(g, h)
```

Опорная идея дальнейшей работы: учет особенностей программы с помощью анализа **графа зависимостей**.
Граф зависимостей этого примера:



Цвет - целевой узел

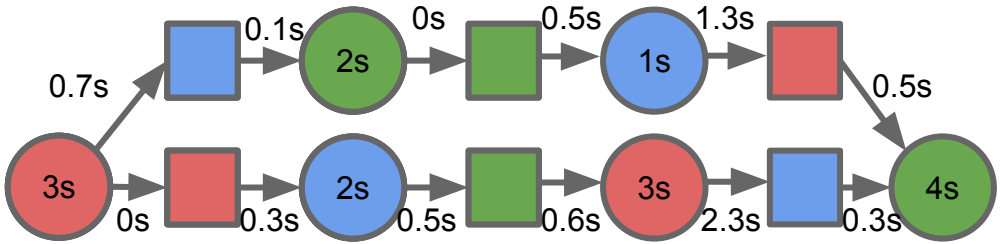
Вспомогательное средство

Оценка стоимости РЗДУ графа зависимостей

Вход:

- Конечный граф зависимостей.
- РЗДУ.
- Вес стоимости каждой задачи и каждого ребра графа зависимостей.
- Детальная конфигурация узлов и топология сети.

Вывод: оценка суммарной стоимости с учетом параллельного исполнения и передач данных.



Предлагаемое решение: анализ конечных графов зависимостей

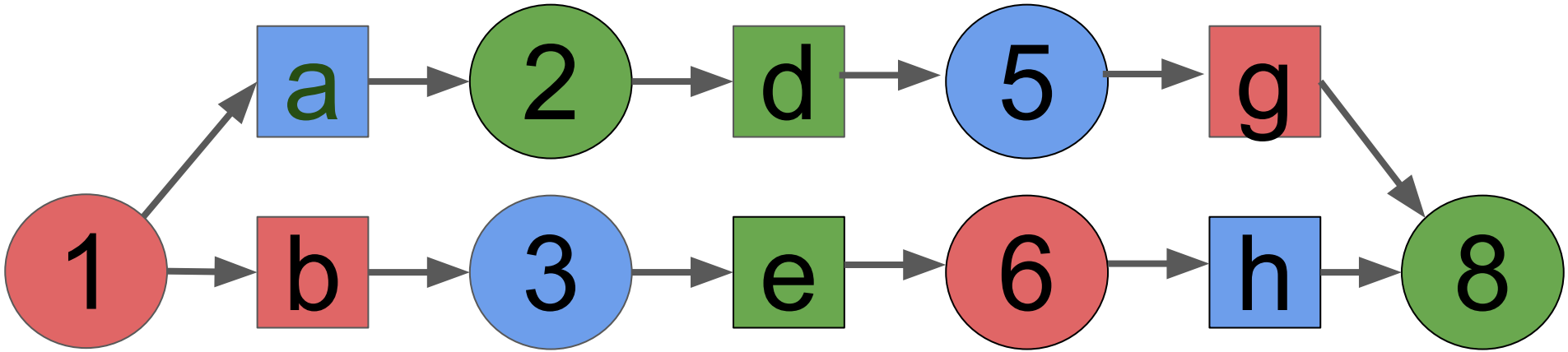
Основные шаги предложенного решения:

1. Раскрутка циклов (loop unrolling) с константным числом итераций, чтобы увеличить количество операторов задач (“строк программы”).
2. Построение **графа зависимостей** полученных операторов задач.
3. **Отображение цепочек** зависимых операторов задач на узлы совместно.
4. Отображение оставшихся операторов задач с помощью приближенной минимизации оценки стоимости РЗДУ на основе генетического алгоритма.

Этот метод хорошо работает, когда в программе много раскручиваемых циклов, среди которых есть параллелизм, заметный на графе зависимостей. Алгоритм учитывает состав вычислительных ядер на каждом узле и неоднородность сети.

Пример работы алгоритма

Исходный граф зависимостей с РЗДУ на основе hash



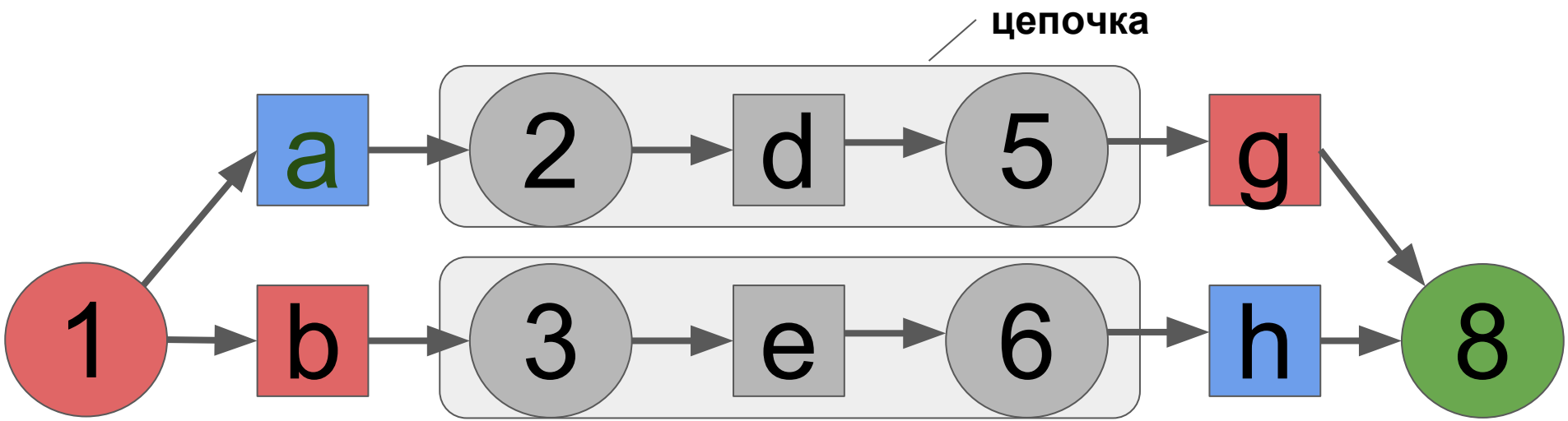
- данные

- задача

Цвет - отображенный узел

Пример работы алгоритма

Отображение цепочек



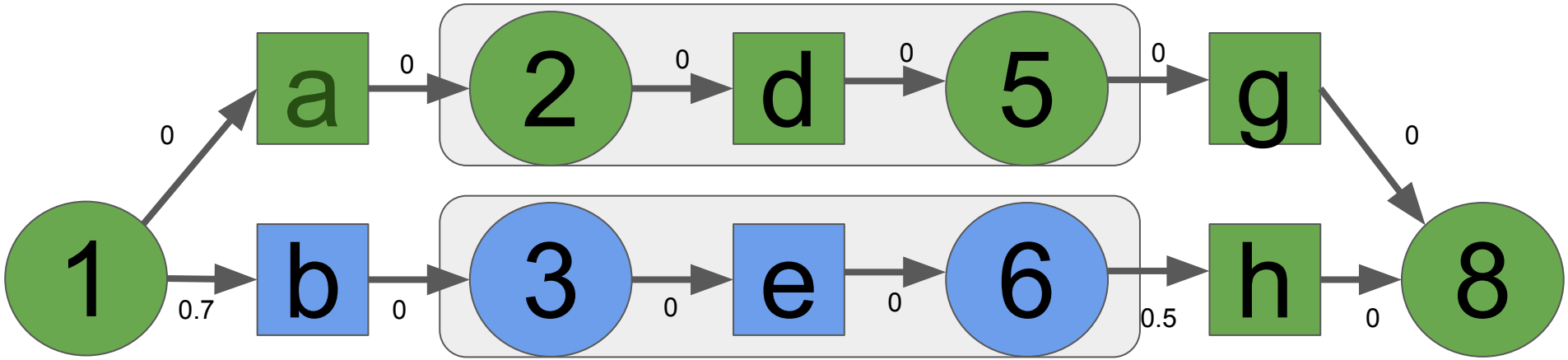
- данные

- задача

Цвет - отображенный узел

Пример работы алгоритма

Приближенная минимизация оценки стоимости РЗДУ

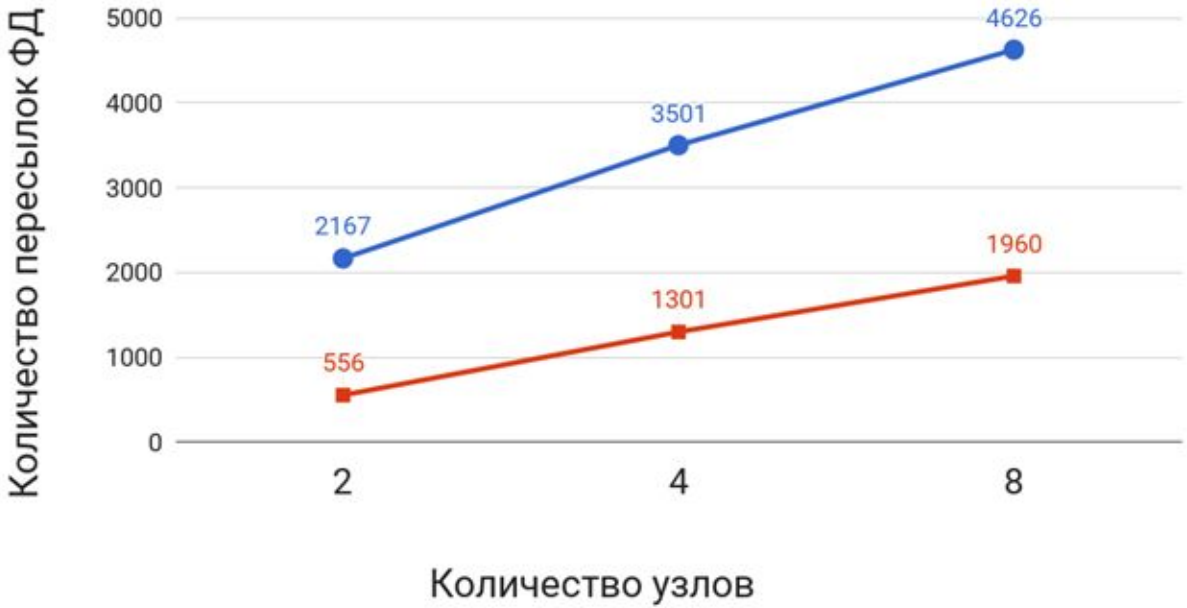


- данные

- задача

Цвет - отображенный узел

Тестирование конструирования РЗДУ на основе поиска цепочек и прототипа генетического алгоритма.
Задача в тестировании - LU-разложение матрицы из 16x16 данных.



Красной линией обозначены результаты реализованного оптимизационного модуля, **синей** - существующего решения на основе hash

Заключение

Выносится на защиту:

- **Аналитический обзор узких мест** алгоритмов LuNA.
- **Разработка** алгоритмов конструирования частных управляющих схем в рамках LuNA, вносящих вклад в решение проблемы **неэффективности** системной части.
- **Реализация и тестирование** частной управляющая схемы конструирования РЗДУ на основе поиска цепочек и прототипа генетического алгоритма.

Дальнейшие планы:

- **Реализация** оставшихся разработанных частных управляющих схем.
- **Разработка** решения без развертки циклов, позволяющее выполнять анализ, зависимый от runtime данных в программе.
- Дальнейший **анализ** предметной области, **разработка** решения других смежных задач.