



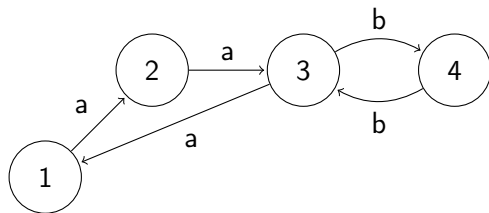
# Параллельная сложность CFL-reachability

Екатерина Шеметова

JetBrains Research, Programming Languages and Tools Lab  
Санкт-Петербургский Академический Университет

14.12.2019

достижимость в графе + контекстно-свободные ограничения



$S \rightarrow aSb \mid ab$

Решения:

$(S, 2, 4), (S, 1, 3), \dots, (S, 3, 3)$

2-4:  $ab$

1-3:  $aabb$

3-4:  $aaabbb$

...

3-3:  $aaaaaabbabbbb$

- Вход: контекстно-свободная грамматика  $G$  и помеченный ориентированный граф  $D$
- Сферы применения: статический анализ кода, запросы к графовым базам данных

- Хотим знать, какие задачи эффективно параллелятся, а какие — нет
- Что в теории значит “эффективно параллеляются”?

- Хотим знать, какие задачи эффективно параллелятся, а какие — нет
- Что в теории значит “эффективно параллеляются”?
  - ▶ Для входа длиной  $n$  задача может быть решена:
    - ★ полиномиальным от  $n$  числом процессоров и
    - ★ за полилогарифмическое от  $n$  время  $\text{polylog}(n)$
  - ▶ Если время хотя бы линейно от входа, то параллелить уже неэффективно

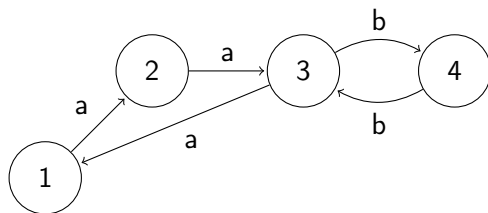
- Классический результат: в общем случае для задачи CFL-reachability нельзя построить эффективный параллельный алгоритм
- Но утверждается, что для некоторых фиксированных классов контекстно-свободных грамматик данная задача эффективно параллелится

- Классический результат: в общем случае для задачи CFL-reachability нельзя построить эффективный параллельный алгоритм
- Но утверждается, что для некоторых фиксированных классов контекстно-свободных грамматик данная задача эффективно параллелится
- **Проблемы и вопросы:** Что объединяет эти эффективные классы? Много ли их? Почему они обладают этим свойством? Можем ли мы получить новые, полезные на практике?

Найти и описать подклассы КС-языков, для которых задача CFL-reachability может быть эффективно решена, а также построить алгоритм, решающий задачу для этих классов.

# Эффективные подклассы

- Эффективность зависит от специального параметра языка — рационального индекса



$S \rightarrow aSb \mid ab$

Решения:

$(S, 2, 4), (S, 1, 3), \dots, (S, 3, 3)$

2-4: ab

1-3: aabb

3-4: aaabbb

...

3-3: aaaaaabbbbbbb



- Эффективность зависит от специального параметра языка — рационального индекса
- Если индекс полиномиален от числа вершин в графе — задача может быть эффективно распараллелена, экспоненциален — нет
- А что влияет на индекс?

# Эффективные подклассы

- Эффективность зависит от специального параметра языка — рационального индекса
- Если индекс полиномиален от числа вершин в графе — задача может быть эффективно распараллелена, экспоненциален — уже нет
- А что влияет на индекс?
  - ▶ Некоторые ограничения на стек pushdown автомата делают рациональный индекс языка полиномиальным!
  - ▶ Примеры: one-turn, oscillation-boundness, один элемент в стековом алфавите и т.д.
- Также полезно изучить влияние операций над языками на величину индекса

- Ограничения на стек pushdown автомата дают нам эффективные языки
- Найдены новые подклассы, для которых можно построить эффективный параллельный алгоритм: металинейные, суперлинейные, oscillation-bounded языки
- А также известные эффективные языки можно комбинировать с помощью операций конкатенации, union, пересечения с регулярными языками, substitution closure и др. и получать новые эффективные примеры
- Для решения задачи адаптирован параллельный алгоритм Брента–Гольдшляггера–Риттера (на эффективных подклассах число процессоров  $O(n^6)$ , время —  $O(\log^2 n)$ ,  $n$  — число вершин в графе)

- **Журнал “Theory of Computing Systems”**

Ekaterina Shemetova, Semyon Grigorev and Alexander Okhotin.

Parallel efficient variants of context-free language reachability problem.

*Статус: планируется публикация*

Прошрое полугодие:

- **Журнал “Труды Института системного программирования РАН”**

Шеметова Е.Н., Григорьев С.В. Задача поиска путей в ациклических графах с ограничениями в терминах булевых грамматик.

*Статус: опубликовано*

- Мы рассматривали сложность для фиксированной грамматики и произвольного графа
- А есть ли эффективные графы для нашей задачи? (кроме тривиальных)
- Планируется изучить влияние структурных характеристик графа, таких как `treewidth`, `pathwidth`, `cutwidth` и др. на эффективность распараллеливания задачи

- Классический алгоритм для решения задачи CFL-reachability работает за  $O(n^3)$  ( $n$  — число вершин в графе).
- Большая открытая проблема: существует ли алгоритм, работающий за  $O(n^{3-\epsilon})$ ?
- Наши текущие результаты: есть субкубическая аппроксимация решения, интересные сведения к другим задачам, также можно использовать результаты про рациональный индекс
- Таких проблем много (пример APSP), и занимается ими целое сообщество (“fine-grained” complexity)
- Ждем ответа от Barna Saha (University of California Berkeley, автор субкубического алгоритма для LED)