Эффективные алгоритмы для CFL-reachability

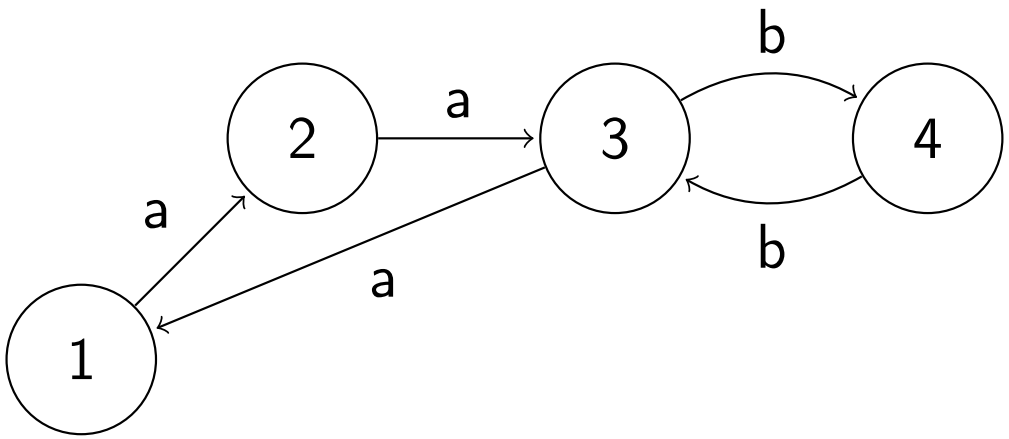
Екатерина Шеметова

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

14.12.2019

CFL-reachability

достижимость в графе + контекстно-свободные ограничения



$S \to aSb \mid ab$

Решения:

$(S, 2, 4), (S, 1, 3), ..., (S, 3, 3)$

2-4: ab

1-3: aabb

3-4: aaabbb

...

3-3: aaaaaabbbbbb

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

Проблемы и цель

Цель: Хотим эффективно решать задачу CFL-reachability

Проблемы:

1. 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 $O(n^{3-\varepsilon})$ (т.н. *субкубический алгоритм*)?

Проблемы и цель

Цель: Хотим эффективно решать задачу CFL-reachability
Проблемы:

1. 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 $O(n^{3-\varepsilon})$ (т.н. *субкубический алгоритм*)?
2. Параллельный алгоритм — доказано, что для задачи CFL-reachability нельзя построить эффективный параллельный алгоритм

Проблемы и цель

Цель: Хотим эффективно решать задачу CFL-reachability
Проблемы:

1. 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 $O(n^{3-\varepsilon})$ (т.н. *субкубический алгоритм*)?
2. Параллельный алгоритм — доказано, что для задачи CFL-reachability нельзя построить эффективный параллельный алгоритм

Подход: Попробуем найти подклассы граматик, для которых задача может быть эффективно решена

Параллельная сложность

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

Параллельная сложность

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

CFL-reachability и параллельная сложность

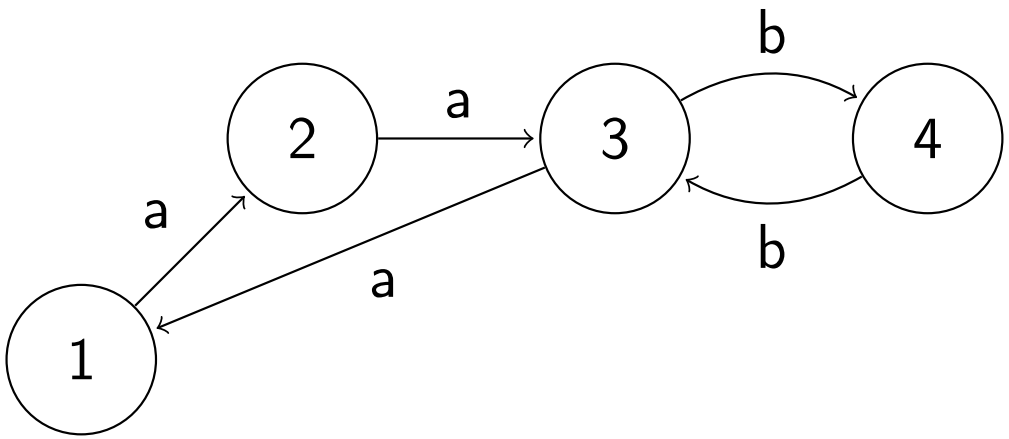
- Классический результат: в общем случае для задачи CFL-reachability нельзя построить эффективный параллельный алгоритм
- Но утверждается, что для некоторых фиксированных классов контекстно-свободных грамматик данная задача эффективно параллелится

CFL-reachability и параллельная сложность

- 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
- **Вопросы:** Что объединяет эти эффективные классы? Много ли их? Почему они обладают этим свойством? Можем ли мы получить новые, полезные на практике?

Эффективные подклассы

- Эффективность зависит от специального параметра языка — рационального индекса $\rho_L(n)$
- $\rho_L(n) = \max\{\min\{|w| : w \in L \cap K\}, K \in Rat_n, L \cap K \neq \emptyset\}$



$S \to aSb \mid ab$

Решения:

$(S, 2, 4), (S, 1, 3), ..., (S, 3, 3)$

2-4: ab

1-3: aabb

3-4: aaabbb

...

3-3: aaaaaabbbbbb

Эффективные подклассы

- Эффективность зависит от специального параметра языка — рационального индекса $\rho_L(n)$
- $\rho_L(n) = \max\{\min\{|w| : w \in L \cap K\}, K \in Rat_n, L \cap K \neq \emptyset\}$
- Если индекс полиномиален от числа вершин в графе — задача может быть эффективно распараллелена, экспоненциален — нет
- Что влияет на величину индекса?

Эффективные подклассы

- Эффективность зависит от специального параметра языка — рационального индекса $\rho_L(n)$
- $\rho_L(n) = \max\{\min\{|w| : w \in L \cap K\}, K \in Rat_n, L \cap K \neq \emptyset\}$
- Если индекс полиномиален от числа вершин в графе — задача может быть эффективно распараллелена, экспоненциален — уже нет
- Что влияет на величину индекса?
  - Некоторые ограничения на стек pushdown автомата делают рациональный индекс оответствующего 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

Результаты

- 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)

Результаты

А что по поводу субкубического алгоритма?

- Эффективные для распараллеливания подклассы языков более эффективны и для последовательного алгоритма
- Благодаря использованию класса one-counter языков, была получена аппроксимация общей задачи за субкубическое время

Публикации

- **Журнал “Theory of Computing Systems”**
  Ekaterina Shemetova, Semyon Grigorev and Alexander Okhotin.
  Parallel efficient variants of context-free language reachability problem.
  Статус: планируется публикация

Прошлое 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

Планы

- 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 решения задачи
- Пробуем использовать найденные эффективные подклассы (как графов, так и грамматик) для получения решения задачи в общем случае
- Подобных проблем много (например APSP), и занимается ими целое сообщество (“fine-grained” complexity). Ждем ответа от Barna Saha (University of California Berkeley, автор субкубического алгоритма для LED).