Дерево доминаторов

Покорский Никита

27 июля 2026

Abstract:

В рамках изучения данной темы я пришёл к выводу, что на просторах русской части интернета нет нормальных конспектов по данной теме, которые были бы понятные школьникам и решил написать его сам.

Keywords:

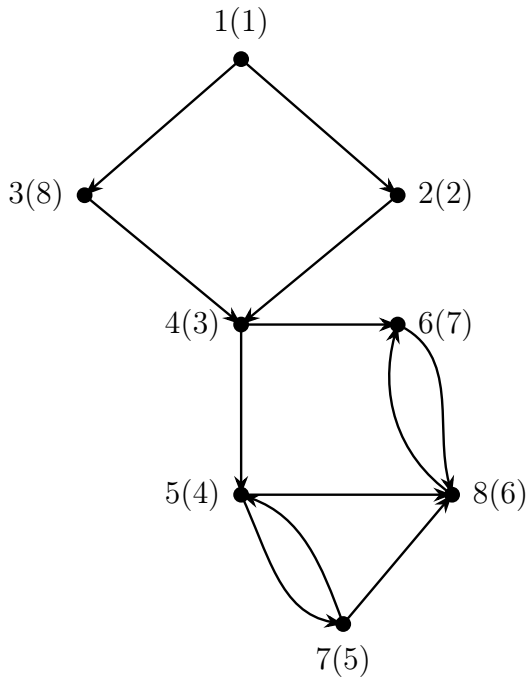
Доминатор, непосредственный доминатор, полудоминатор, относительный доминатор, теорема о доминаторах, быстрый поиск дерева доминаторов.

Теоретическая часть

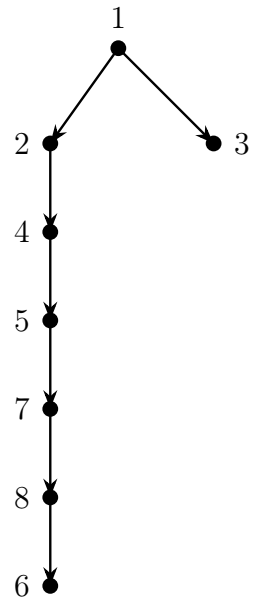
Пусть нам изначально задан ориентированный граф $G = (V, E)$ и некоторая вершина s в нём, являющаяся корнем. Кроме того, положим, что все вершины $v \in V$ достижимы из s.

Доминатором вершины v будем называть вершину $k \neq v$, такую, что для всякого пути $s \overset{p}{\rightsquigarrow} v$ $k \in p$. Далее будем записывать это как $k \in \mathrm{dom}(v)$.

Давайте запустим DFS из вершины r и переупорядочим вершины в каком-нибудь порядке времени входа в неё. Далее для удобства будем говорить, что новый номер вершины v после нашей процедуры станет v^*. Также будем обозначать, что u — предок v в дереве обхода DFS, как $u \prec v$. Кроме того, если u встретилась раньше чем v, то будем это записывать как $u \underset{tin}{<} v$:



(а) Исходный граф



(б) Дерево обхода DFS

Рис. 1: Переупорядочивание вершин графа.

Лемма 1: Необходимое условие доминирования

Если вершина k является доминатором вершины v, то k — предок вершины v в $DFST$. Более формально, $k \prec v$.

Доказательство. Очевидно. □

Кроме того, давайте выпишем некоторые свойства доминаторов.

1. *Антисимметричность*: иными словами, не может быть такого, что u доминатор v и v доминатор u.

2. *Транзитивность*: если u доминатор v и v доминатор k, то u доминатор k.

3. *Полнота*: если u и v доминаторы w, то либо u доминатор v, либо наоборот.

Доказательство. (1) Очевидно следует из леммы 1.

(2) Если всякий путь из s в v содержит вершину u и из s в k содержит вершину v, то неизбежно все пути из s в k содержат вершину u.

(3) Предположим противное: пусть у нас обе вершины являются доминаторами w, но ни одна из них не доминирует над другой.

Тогда существует путь (4): $s \overset{p}{\rightsquigarrow} v$, $u \notin p$. Однако из того, что они обе являются доминаторами вершины w, следует, что всякий путь от s до w проходит через u и v. Выберем произвольный; не умаляя общности, он имеет вид $s \rightsquigarrow u \rightsquigarrow v \rightsquigarrow w$. Тогда в силу (4) имеем, что можно было исключить u из этого пути, а значит, она не доминатор w. □

В силу свойств, описанных выше, для каждой вершины можно определить новое понятие.

Непосредственный доминатор вершины v — это такая вершина k, что одновременно выполняются следующие условия:

$$\begin{cases} \mathrm{k} \in \mathrm{dom(v)}, \\ \nexists t \in V: \ t \in \mathrm{dom}(v), k \in dom(t) \end{cases}.$$

Далее будем записывать это как $k = \mathrm{idom}(v)$.

Стоит отметить, что для каждой вершины, кроме быть может корня, непосредственный доминатор определён однозначно.

Доказательство. Прямое следствие из полноты отношения доминирования. □

Введём последнее определение и перейдём к некоторой практической части.

Дерево доминаторов графа G будем называть дерево T, рёбрами которого являются $(\mathrm{idom}(v), v)$. Помимо того, зафиксируем его корень – s.

Поиск дерева доминаторов

Нам дан ориентированный граф $G = (V, E, s)$, мы хотим для него построить дерево доминаторов. Наивный алгоритм: давайте удалим вершину v из него вместе со всеми рёбрами, инцидентными ей. Тогда для всех непосещённых вершин наша вершина v является доминатором. Далее, если построить граф G', в котором будем проводить ребро $(u, v) \Leftrightarrow u \in \mathrm{dom}(v)$, тогда G' – DAG, а значит, на нём можно построить топологическую сортировку. Будем говорить, что непосредственным доминатором вершины v является вершинка k, если её номер наибольший в топологической сортировке среди доминаторов v. Такой алгоритм работает за $\mathcal{O}(VE)$.

Давайте чуть более подробно исследуем структуру непосредственных доминаторов.

Полудоминатор вершины v — это такая вершинка k, что её номер после перенумерации минимальный, а также существует путь в G из $k \rightsquigarrow v$, что номера всех вершин в G' на этом пути, кроме, быть может, v, строго больше, чем у k. Более формально, $k - \underset{tin}{\mathrm{argmin}}\ t \in V : \exists t \overset{p}{\rightsquigarrow} v, \forall u \in p \setminus \{t, v\}\ u \underset{tin}{>} v$. Далее будем говорить, что $k = \mathrm{sdom}(v)$.

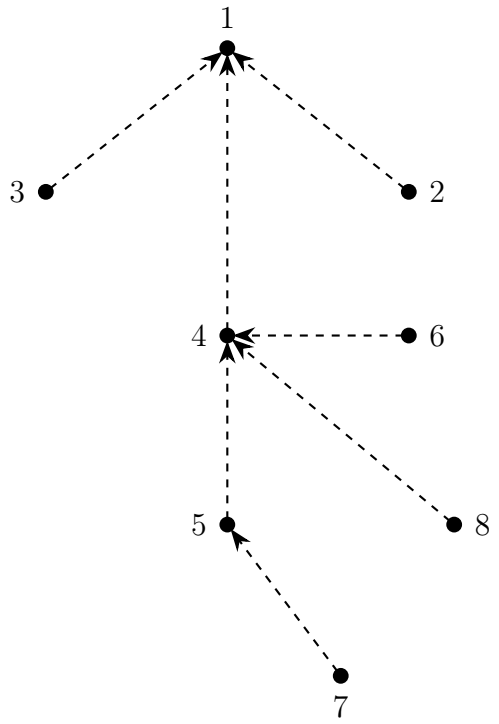


Рис. 2: Полудоминаторы в исходном графе.

Поскольку $p(v)$ — кандидат на $\mathrm{sdom}(v)$, то $\mathrm{sdom}(v) \underset{tin}{<} v$.

Лемма 2: Положение $\mathrm{sdom}(v)$ относительно v в $DFST$

$\mathrm{sdom}(v)$ лежит на пути до корня от вершины v в дереве обхода DFS. Иными словами, $\mathrm{sdom}(v) \prec v$.

Доказательство. Пусть $u = \mathrm{sdom}(v)$. Из замечания выше мы знаем, что $u \prec v$. Кроме того, по определению $\mathrm{sdom}(v)$ у нас есть путь:

$$u = x_0 \rightsquigarrow \cdots \rightsquigarrow x_{k+1} = v, \forall i \in \overline{1 \dots k} : x_i \underset{tin}{>} v$$

Тогда в момент, когда мы войдём в u, все вершины пути будут ещё не помечены. С другой стороны, в силу свойств обхода DFS он обойдёт все вершины, достижимые из u, перед тем как выйти из неё, а значит, v лежит в поддереве u. □

Относительным доминатором вершины v будем называть вершину k, отличную от $\mathrm{sdom}(v)$, лежащую на пути в DFST $\mathrm{sdom}(v) \overset{p}{\rightsquigarrow} v$, такую, что $\forall x \in p \setminus \mathrm{sdom}(v) : \mathrm{sdom}(k) \underset{tin}{\leq} \mathrm{sdom}(x)$. Далее будем её обозначать как $k = \mathrm{rdom}(v)$.

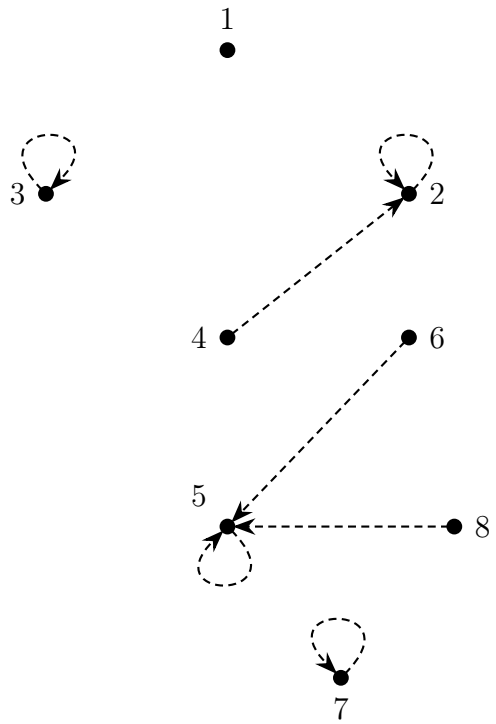


Рис. 3: Относительные доминаторы в исходном графе.

По Лемме 2 относительный доминатор лежит на пути от вершины v до корня.

Лемма 3: Положение $\mathrm{idom}(v)$ относительно $\mathrm{sdom}(v)$ в $DFST$

$\mathrm{idom}(v)$ является предком $\mathrm{sdom}(v)$ в дереве обхода DFS.

Доказательство. Пусть $u = \mathrm{sdom}(v)$, $k = \mathrm{idom}(v)$.

Мы знаем, что $\mathrm{idom}(v)$ – предок v и $\mathrm{sdom}(v)$ – предок v (лемма 2). Мы хотим показать, что случай $s \underset{T}{\rightsquigarrow} \mathrm{sdom}(v) \underset{T}{\rightsquigarrow} \mathrm{idom}(v) \underset{T}{\rightsquigarrow} v$ невозможен. В таком случае единственный доступный вариант, что $\mathrm{idom}(v)$ – предок $\mathrm{sdom}(v)$ в DFST.

Предположим противное: тогда $u \prec k \prec v$. Тогда в силу того, что $u = \mathrm{sdom}(v)$, есть путь до v, который идёт по вершинам, чей tin строго больше v; с другой стороны, $k \prec v$. Тогда $\exists p, s \rightsquigarrow u \overset{p}{\rightsquigarrow} v$, что в нём не будет вершины k. Значит, k не доминатор v. □

Давайте теперь ввдеем главную теорему, на которую будет опираться наш алгоритм, а потом перейдем к практической части.

Теорема 1: Теорема о доминаторах

Для всякой вершины графа, кроме, быть может, s, выполнено одно из следующих условий:

1. Если $\mathrm{rdom}(v) = v$, то $\mathrm{idom}(v) = \mathrm{sdom}(v)$.
2. Иначе $\mathrm{idom}(v) = \mathrm{idom}(\mathrm{rdom}(v))$.

Доказательство. Покажем, что если $\mathrm{sdom}(v) \neq \mathrm{idom}(v)$, то $\mathrm{rdom}(v) \neq v$. В противном случае $\mathrm{idom}(v) = \mathrm{idom}(\mathrm{rdom}(v))$.

(1) Для доказательства первой части рассмотрим путь: $s \overset{p}{\rightsquigarrow} v, \mathrm{sdom}(v) \notin p$ (имеем право, так как $\mathrm{sdom}(v) \neq \mathrm{idom}(v)$, кроме того, в силу леммы 3 он идет после $\mathrm{idom}(v)$). Пусть x — это последняя вершина на пути, такая, что $x \underset{tin}{<} \mathrm{sdom}(v)$, а y – это минимальная вершина на пути p между x и v. Тогда для вершины y x – кандидат на $\mathrm{sdom}(y)$. С другой стороны, в силу выбора x мы получаем неравенство: $\mathrm{sdom}(y) \underset{tin}{<} x \underset{tin}{<} sdom(v)$, тогда $\mathrm{rdom}(v) \neq v$. Тем самым мы завершили первую часть доказательства.

(2) Осталось показать, что в противном случае $\mathrm{idom}(v) = \mathrm{idom}(\mathrm{rdom}(v))$. По замечанию об относительном доминаторе вершины v всякий путь до $\mathrm{rdom}(v)$ можно продлить до v; иными словами, существует рёберно-простой путь в дереве вида $s \rightsquigarrow \mathrm{rdom}(v) \rightsquigarrow v$. Положим, что $\mathrm{idom}(v) \neq \mathrm{idom}(\mathrm{rdom}(v))$. Тогда существует путь $p : s \overset{p}{\rightsquigarrow} v$, что $\mathrm{idom}(\mathrm{rdom}(v)) \notin p$. Пусть x — это последняя вершина на нём, такая, что $x \underset{tin}{<} \mathrm{idom}(\mathrm{rdom}(v))$, а y – это минимальная вершина на пути p между x и v. Тогда, с одной стороны, x – это кандидат на $\mathrm{sdom}(y)$; с другой стороны, y

является предком v в $DFST$, тогда:

$$(1)\ \ \mathrm{sdom}(y) \underset{tin}{\leq} x \underset{tin}{<} \mathrm{idom}(\mathrm{rdom}(v))$$

.

Поскольку y находится на древесном пути к v, а $\mathrm{rdom}(v)$ выбирается как вершина с минимальным $\mathrm{sdom}(x)$ на этом пути, то получаем неравенство:

$$(2)\ \ \mathrm{sdom}(\mathrm{rdom}(v)) \underset{tin}{\leq} \mathrm{sdom}(y)$$

Объединив (1) и (2) получаем неравенство вида:

$$\mathrm{sdom}(\mathrm{rdom}(v)) \underset{tin}{<} \mathrm{idom}(\mathrm{rdom}(v))$$

Противоречие с Леммой 3 (О взаиморасположении $\mathrm{idom}(v)$ и $\mathrm{sdom}(v)$). Тем самым завершаем вторую часть доказательства.

□

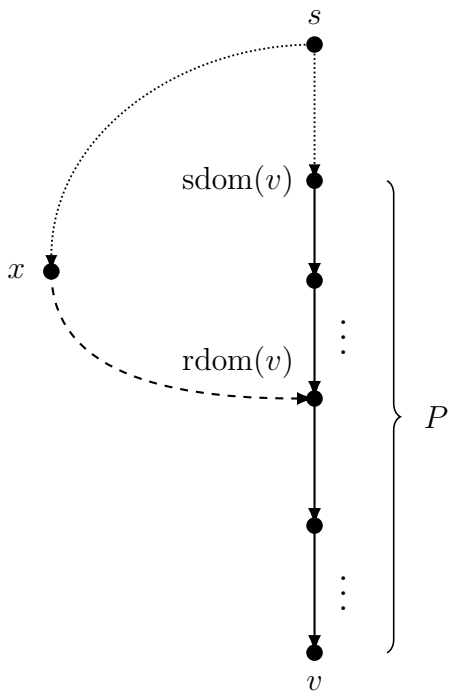


Рис. 4: Случай $\mathrm{rdom}(v) \neq v$. P – древестный путь между $\mathrm{sdom}(v)$ и v.

Быстрый поиск дерева доминаторов

В данном разделе будет изложен не самый быстрый способ поиска, работающий за $\mathcal{O}((m+n)\log n)$, однако известен алгоритм, работающий за $\mathcal{O}(m\alpha(m,n)+n)$. И даже за $\mathcal{O}(n+m)$.

Общая схема нашего алгоритма:

1. Построить обход DFS.
2. Насчитать для каждой вершины $v \neq s$: $\mathrm{sdom}(v)$.
3. Насчитать для каждой вершины $v \neq s$: $\mathrm{rdom}(v)$.
4. В обратном порядке обхода DFS для каждой вершины найти $\mathrm{idom}(v)$.

Для решения пункта (2) введём вспомогательную лемму и несколько обозначений:

Рассмотрим некоторое ребро $(u,v) \in E$ из нашего графа, пусть $t = \mathrm{lca}(u,v)$. Если $u = t$, то $\mathrm{r}(u,v) = u$, в противном случае рассмотрим путь:

$$t = x_0 \rightsquigarrow x_1 \rightsquigarrow \ldots \rightsquigarrow x_l = u$$

Тогда $\mathrm{r}(u,v) = k$, такой, что $k \in \{x_1, \ldots, x_l\}$ и $\forall i \in \overline{1 \ldots l} : \mathrm{sdom}(x_i) \underset{tin}{\geq} \mathrm{sdom}(k)$. Прошу заметить, что мы не рассматриваем вершину t на этом пути в качестве кандидата на $\mathrm{r}(u,v)$. Более формально:

$$\mathrm{r}(u,v) = \begin{cases} u, u = \mathrm{lca}(u,v) \\ \underset{tin}{\mathrm{argmin}} \left(\mathrm{sdom}(x_i) \;\middle|\; x_i \in t \underset{T}{\rightsquigarrow} u \setminus \{t\} \right), u \neq \mathrm{lca}(u,v) \end{cases}$$

Лемма 4: Лемма о $\mathrm{sdom}(v)$

$\mathrm{sdom}(v)$ есть не что иное, как вершина с наименьшим значением tin среди всех вершин r(u,v), соответствующих рёбрам, входящим в v. Более формально это можно записать следующим образом:

$$\mathrm{sdom}(v) = \underset{tin}{\mathrm{argmin}} \big(x \;\big|\; \exists (u,v) \in E, x = \mathrm{r}(u,v) \big)$$

.

Доказательство. Поскольку вершина $x = \mathrm{r}(u, v)$ имеет минимальный tin по всем допустимым путям, оканчивающимся данным ребром, то минимальная вершина t по всем таким вершинам x удовлетворяет условию: $t \underset{tin}{\leq} \mathrm{sdom}(v)$. Но по определению $\mathrm{sdom}(v)$ имеем: $\mathrm{sdom}(v) \underset{tin}{\leq} t$. Тогда $t = \mathrm{sdom}(v)$. □

Пользуясь данной леммой, мы сразу получаем алгоритм поиска $\mathrm{sdom}(v)$:

Алгоритм 1: Алгоритм поиска $\mathrm{sdom}(v)$

Давайте идти в обратном порядке обхода DFS. Пусть мы хотим найти $\mathrm{sdom}(v)$, тогда нам достаточно рассмотреть все входящие в v рёбра (u, v). Каждое такое ребро задаёт кандидата на $\mathrm{sdom}(v)$ – $\mathrm{r}(u, v)$. Тогда $\mathrm{sdom}(v) = \underset{tin}{\mathrm{argmin}}(\mathrm{r}(u, v) \mid (u, v) \in E)$. Для эффективного поиска $\mathrm{r}(u, v)$ можно хранить структуру данных, которая будет поддерживать сжатые пути из дерева DFS. На каждом сжатом пути она будет хранить вершину x, для которой значение

$$tin(sdom(x)) \text{ – минимально.}$$

Несложно видеть, что для этого хватит СНМ с эвристикой сжатия путей.

Доказательство. Поскольку вершины обрабатываются в обратном порядке DFS, то все необходимые значения $\mathrm{sdom}(x)$ к этому моменту уже вычислены. □

С псевдокодом можно ознакомиться в разделе Приложения.

Для решения пункта (3) достаточно явно построить $DFST$ и на нём предподсчитать бинапы, на каждом из которых хранить вершину x, для которой $\mathrm{sdom}(x)$ имеет минимальный tin на бинапе.

Для решения пункта (4) достаточно пройтись в прямом порядке DFS по вершинам и, пользуясь теоремой о доминаторах, насчитать $\mathrm{idom}(v)$ для всех вершин.

Приложения

Result: sdom
// $p[v]$ - предок вершины v в дереве DFS
// $par[v]$ - предок вершины v в сжатом дереве
// $path\_min[v]$ - вершина с минимальным tin на сжатом пути от v
for $v \in V$ **do**
  $par[v] \leftarrow 0$;
end
for $v \in$ *reverse DFS order* **do**
  $sdom[v] \leftarrow \operatorname{argmin}_{tin}(\mathrm{chk}(u) \mid (u, v) \in E)$;
  $par[v] \leftarrow p[v]$;
  $path\_min[v] \leftarrow sdom[v]$;
end
return sdom;

Algorithm 1: Поиск полудоминаторов для вершин графа G.

Result: $\mathrm{r}(u, v)$
Function *chk* (v)
  if $par[v] = 0$ **then**
    return v;
  end
  if $par[par[v]] \neq 0$ **then**
    $path\_min[v] = \operatorname{argmin}_{tin}(path\_min[v], chk(par[v]))$;
    $par[v] = par[par[v]]$;
  end
  return $path\_min[v]$;
end

Algorithm 2: Сжатие пути и поиск минимума на нём.