10.5.2. Поиск с возвратами

Даже если для решения задач, подобных поставленной в предыдущем подразделе, не удаётся найти эффективного алгоритма, остаётся возможность попробовать найти решение «полным перебором» всех возможных вариантов, просто в силу конечности числа возможностей. Например, наибольшее независимое множество можно найти по следующей схеме.

Вход: граф $G(V, E)$.
Выход: наибольшее независимое множество X.

$m := 0$ { наилучшее известное значение β_0 }
for $Y \in 2^V$ **do**
  if $Y \in \mathcal{E}$ & $|Y| > m$ **then**
    $m := |Y|; X := Y$ { наилучшее известное значение X }
  end if
end for

ЗАМЕЧАНИЕ

Для выполнения этого алгоритма потребуется $O(2^p)$ шагов.

ОТСТУПЛЕНИЕ

Алгоритм, трудоёмкость которого (число шагов) ограничена полиномом от характерного размера задачи, принято называть *эффективным*, в противоположность *неэффективным* алгоритмам, трудоёмкость которых ограничена функцией, растущей быстрее, например, экспонентой. Таким образом, жадный алгоритм эффективен, а полный перебор — нет.

При решении переборных задач большое значение имеет способ организации перебора (в нашем случае — способ построения и последовательность перечисления множеств Y). Наиболее популярным является следующий способ организации перебора, основанный на идее поиска в глубину и называемый *поиском с возвратами*.

ЗАМЕЧАНИЕ

Иногда употребляется термин «*бэктрекинг*» (транслитерация английского названия этого метода — *backtracking*).

Идея поиска с возвратами состоит в следующем. Находясь в некоторой ситуации, пробуем изменить её допустимым образом в надежде найти решение. Если изменение не привело к успеху, то возвращаемся в исходную ситуацию (отсюда название «поиск с возвратами») и пробуем изменить её другим образом, и так до тех пор, пока не будут перебраны все возможности.

Для рассматриваемой задачи отыскания наибольшего независимого множества вершин метод поиска с возвратами может быть реализован с помощью следующего рекурсивного алгоритма.

Алгоритм 10.2. Поиск с возвратами

Вход: граф $G(V, E)$.
Выход: наибольшее независимое множество X.
$m := 0$ { наилучшее известное значение β_0 }
$X := \varnothing$ { наибольшее известное независимое множество X }
$\mathrm{BT}(\varnothing, V)$ { вызов рекурсивной процедуры BT }

Основная работа выполняется рекурсивной процедурой BT.

Вход: S — текущее независимое множество вершин, T — оставшиеся вершины графа.
Выход: изменение глобальной переменной X, если текущее множество не может быть расширено (является максимальным).
if $|S| > m$ **then**
$X := S; m := |S|$ { наибольшее известное независимое множество }
end if
for $v \in T$ **do**
if $S + v \in \mathcal{E}$ **then**
$\mathrm{BT}(S + v, T \setminus \Gamma^*(v))$ { пробуем добавить v }
end if
end for

Обоснование. По построению вершина v добавляется в множество S только при сохранении независимости расширенного множества. В алгоритме это обстоятельство указано в форме условия $S + v \in \mathcal{E}$. Проверить сохранение условия независимости нетрудно, например, с помощью следующей функции.

Вход: независимое множество S и проверяемая вершина v.
Выход: **true**, если множество $S + v$ независимое, **false** — в противном случае.
for $u \in S$ **do**
if $(u, v) \in E$ **then**
return false { множество $S + v$ зависимое }
end if
end for
return true { множество $S + v$ независимое }

Этот цикл не включен в явном виде в рекурсивную процедуру BT, чтобы не загромождать основной текст и не затуманивать идею поиска с возвратами. Таким образом, множество S, а следовательно, и множество X — независимые. В тот момент, когда множество S нельзя расширить, оно максимально по определению. Переменная m глобальна, поэтому среди всех максимальных независимых множеств в конце работы алгоритма построенное множество X является наибольшим независимым множеством вершин. □