IQRO JURNALI , IQRO JOURNAL, ИКРО ЖУРНАЛИ
ISSN:2181-4341| wordlyknowledge.uz/index.php/iqro
Volume 21 Issue 02 May 2026

ЭКСТРЕМАЛЬНЫЕ ЗАДАЧИ НА ГРАФАХ

Эшниязов Абдумалик Искандарович

д.ф.ф.-м.н. (PhD), доцент

Холматов Кахрамон Холмуротбой угли Магистр

Doi: https://doi.org/10.5281/zenodo.20356721

Аннотация: В статье предложены основные факты, касающиеся экстремальных задач теории графов, приведены основные теоремы с доказательствами. Показана достижимость оптимальных решений этих задач в рамках фрагментарной модели. Предложены приближенные алгоритмы поиска решений методом Прима.

Ключевые слова: Независимые множества, клики, вершинные покрытия, паросочетания, оптимальные каркасы, алгоритм Прима.

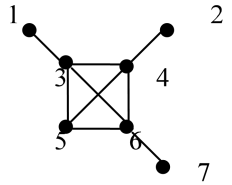
Abstract: The article proposes the main facts concerning extremal problems of graph theory, gives the main theorems with proofs. The attainability of optimal solutions to these problems within the framework of a fragmentary model is shown. Approximate algorithms for finding solutions by the Prim method are proposed.

Keywords: Independent sets, cliques, vertex covers, matchings, optimal skeletons, Prim's algorithm.

Независимым множеством вершин графа называется любое множество попарно не смежных вершин, т.е. множество вершин, порождающее пустой подграф. Независимое множество называется *максимальным*, если оно не является собственным подмножеством другого независимого множества, и *наибольшим*, если содержит наибольшее количество вершин. Число вершин в наибольшем независимом множестве графа G обозначается через $\alpha(G)$ и называется *числом независимости* графа. Задача о независимом множестве состоит в нахождении наибольшего независимого множества .

Кликой графа называется множество вершин, порождающее полный подграф, т.е. множество вершин, каждые две из которых смежны. Число вершин в клике наибольшего размера называется *кликовым числом* графа и обозначается через $k(G)$. Очевидно, задача о независимом множестве трансформируется в задачу о клике и наоборот простым переходом от данного графа G к дополнительному графу $\overline{G}$, так что $\alpha(G) = k(\overline{G})$.

Вершинное покрытие графа – это такое множество вершин, что каждое ребро графа инцидентно хотя бы одной из этих вершин. Наименьшее число вершин в вершинном покрытии графа G обозначается через $\beta(G)$ и называется *числом вершинного покрытия* графа.



В графе на рисунке 1 наибольшим независимым множеством является множество {1,2,5,7}, наибольшей кликой – множество {3,4,5,6}, наименьшим вершинным покрытием – множество {3,4,6}.

Между задачами о независимом множестве и о вершинном покрытии тоже имеется простая связь благодаря следующему легко устанавливаемому факту.

Теорема 1. *Подмножество U множества вершин графа G является вершинным покрытием тогда и только тогда, когда $VG - U$ – независимое множество.*

В частности, отсюда следует, что $\alpha(G) + \beta(G) = n$ для любого графа G с n вершинами.

Таким образом, все три задачи тесно связаны друг с другом. Далее рассмотрим задачу о независимом множестве.

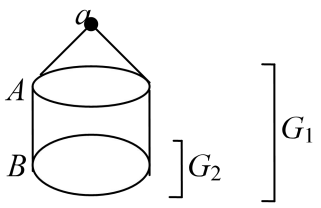
Пусть G – граф, в котором требуется найти наибольшее независимое множество. Выберем в нем произвольную вершину a и пусть A – множество всех вершин графа, смежных с a, B – множество всех вершин, не смежных с a. Обозначим через G_1 подграф, получающийся удалением из графа G вершины a, а через G_2 подграф, получающийся удалением из G всех вершин множества $A \quad \{a\}$. Иначе говоря, G_1 – подграф графа G, порожденный множеством $A \quad B$, а G_2 – подграф, порожденный множеством B (см. рисунок 2).

Рис. 2

Пусть X – какое-нибудь независимое множество графа G. Если оно не содержит вершину a, то оно является независимым множеством графа G_1, если же $a \quad X$, то множество $X - \{a\}$ является независимым множеством графа G_2. Таким образом, задача о независимом множестве для графа G свелась к решению той же задачи для двух графов меньшего размера. Это приводит к рекуррентному уравнению для числа независимости:

$$\alpha(G) = \max\{\alpha(G_1), \alpha(G_2) + 1\}$$

и к рекурсивному алгоритму для нахождения наибольшего независимого множества графа G: найдем наибольшее независимое множество X_1 графа G_1, затем наибольшее независимое множество X_2 графа G_2 и выберем большее из множеств X_1 и $X_2 \quad \{a\}$. В целом процесс решения задачи можно при этом интерпретировать как исчерпывающий поиск в возникающем дереве подзадач, т.е. подграфов. Каждой вершине этого дерева соответствует некоторый граф, сыновьям вершины – два подграфа этого графа, корню – исходный граф. Листьям дерева соответствуют графы, для которых задача решается непосредственно, без сведения к подзадачам. В простейшем случае это могут быть графы с пустым множеством вершин, но при этом количество листьев будет максимальным. Можно доводить разложение на подзадачи до графов с пустым множеством ребер, при этом листьев в дереве станет меньше, но потребуется для каждого встречающегося графа



проверять наличие в нем ребер. Число листьев зависит от того, в каком порядке рассматриваются вершины графа, но оно во всяком случае не меньше, чем число максимальных независимых множеств у графа, так как каждое из этих множеств будет соответствовать некоторому листу. Так, для графа pK_2, т.е. графа, состоящего из p компонент связности, каждая из которых изоморфна графу K_2, в дереве подзадач будет в лучшем случае 2^p листьев.

Паросочетанием в графе называется множество ребер, попарно не имеющих общих вершин. Задача о паросочетании состоит в том, чтобы в данном графе найти паросочетание с наибольшим числом ребер. Это число для графа G будем обозначать через $\pi(G)$. *Реберным покрытием* графа называется такое множество ребер, что всякая вершина графа инцидентна хотя бы одному из этих ребер. Наименьшее число ребер в реберном покрытии графа G обозначим через $\rho(G)$. Заметим, что реберное покрытие существует только для графов без изолированных вершин.

Определение паросочетания похоже на определение независимого множества вершин, паросочетание иногда так и называют – независимое множество ребер. Эта аналогия усиливается еще тесной связью между реберными покрытиями и паросочетаниями, подобно тому, как связаны между собой вершинные покрытия и независимые множества. Даже равенство, количественно выражающее эту связь, имеет точно такой же вид (напомним, что числа независимости $\alpha(G)$ и вершинного покрытия $\beta(G)$ связаны равенством $\alpha(G)+\beta(G)=n$). Приведем доказательство этого факта, так как оно имеет алгоритмическое значение – показывает, как каждая из двух задач сводится к другой.

Теорема 2. *Для любого графа G с n вершинами, не имеющего изолированных вершин, справедливо равенство* $\pi(G)+\rho(G)=n$.

Доказательство. Пусть M – наибольшее паросочетание в графе G. Обозначим через W множество всех вершин графа, не покрытых ребрами этого паросочетания. Тогда $|W|=n-2\pi(G)$. Очевидно, W – независимое множество (иначе M не было бы наибольшим). Выберем для каждой вершины из M какое-нибудь инцидентное ей ребро. Пусть F – множество всех выбранных ребер. Тогда M F – реберное покрытие и $|M$ $F|=n-\pi(G)$, следовательно, $\rho(G)$ $n-\pi(G)$.

Обратно, пусть C – наименьшее реберное покрытие графа G. Рассмотрим подграф H графа G, образованный ребрами этого покрытия. В графе H один из концов каждого ребра является вершиной степени 1 (ребро, каждая вершина которого инцидентна по крайней мере еще одному ребру, можно было бы удалить из C, оставшиеся ребра по-прежнему покрывали бы все вершины). Отсюда следует, что каждая компонента связности графа H является звездой (звезда – это дерево, у которого не более одной вершины степени больше 1). Так как в любом лесе сумма количеств ребер и компонент связности равна числу вершин, то число компонент связности в графе H равно $n-\rho(G)$. Выбрав по одному ребру из каждой компоненты, получим паросочетание. Отсюда следует, что $\pi(G)$ $n-\rho(G)$.

Пусть G – граф, M – некоторое паросочетание в нем. Ребра паросочетания будем называть *сильными,* остальные ребра графа – *слабыми*. Вершину назовем *насыщенной,* если она инцидентна какому-либо ребру паросочетания, в противном случае *ненасыщенной*,. На рисунке 3 слева показан граф и в нем выделены ребра паросочетания $M = \{(2,3),(4,5),(7,8)\}$. Вершины 1 и 5 – ненасыщенные. Заметим, что к этому паросочетанию нельзя добавить ни одного ребра, т.е. оно максимальное. Однако оно не является наибольшим. В этом легко убедиться, если рассмотреть путь 5,6,8,9,10,7,3,4 (показан пунктиром). Он начинается и оканчивается в ненасыщенных вершинах, а вдоль пути чередуются сильные и слабые ребра. Если на этом пути превратить каждое сильное ребро в слабое, а каждое слабое – в сильное, то получится новое паросочетание, показанное на рисунке справа, в котором на одно ребро больше. Увеличение паросочетания с помощью подобных преобразований – в этом и состоит суть метода чередующихся цепей.



Рис.3

Сформулируем необходимые понятия и докажем теорему, лежащую в основе этого метода чередующихся цепей. *Чередующейся цепью* относительно данного паросочетания называется простой путь, в котором чередуются сильные и слабые ребра (т.е. за сильным ребром следует слабое, за слабым – сильное). Чередующаяся цепь называется *увеличивающей*, если она соединяет две ненасыщенные вершины. Если M – паросочетание, P – увеличивающая цепь относительно M, то легко видеть, что $M \quad P$ – тоже паросочетание и $|M \quad P| = |M| + 1$.

Теорема 3. *Паросочетание является наибольшим тогда и только тогда, когда относительно него нет увеличивающих цепей.*

Доказательство. Если есть увеличивающая цепь, то, поступая так, как в рассмотренном примере, т.е. заменяя вдоль этой цепи сильные ребра на слабые и наоборот, мы, очевидно, получим большее паросочетание. Для доказательства обратного утверждения рассмотрим паросочетание M в графе G и предположим, что M не наибольшее. Покажем, что тогда имеется увеличивающая цепь относительно M. Пусть M – другое паросочетание и $|M| > |M|$. Рассмотрим подграф H графа G, образованный теми ребрами, которые входят в одно и только в одно из паросочетаний M, M. Иначе говоря, множеством ребер графа H является симметрическая разность $M \quad M$. В графе H каждая вершина инцидентна не более чем двум ребрам (одному из M и одному из M), т.е. имеет степень не более двух. В таком графе каждая компонента связности – путь или цикл. В каждом из этих путей и циклов чередуются ребра из M и M. Так как $|M| > |M|$, то имеется компонента, в которой ребер из M содержится больше, чем ребер из M. Это может быть только путь, у которого оба концевых ребра принадлежат M. Легко видеть, что относительно M этот путь будет увеличивающей цепью.

Для решения задачи о паросочетании остается научиться находить увеличивающие цепи или убеждаться, что таких нет. Тогда, начиная с любого паросочетания (можно и с пустого множества ребер), можем строить паросочетания со все увеличивающимся количеством ребер до тех пор, пока не получим такое, относительно которого нет увеличивающих цепей. Оно и будет наибольшим. Известны эффективные алгоритмы, которые ищут увеличивающие цепи для произвольных графов, но они достаточно сложны. Рассмотрим более простой алгоритм, решающий эту задачу для двудольных графов.

Пусть $G = (A, B, E)$ – двудольный граф, M – паросочетание в G, X и Y – множества ненасыщенных вершин соответственно в долях A и B. Всякая увеличивающая цепь, если такая имеется, соединяет вершину из A с вершиной из B. Ориентируем ребра графа G следующим образом: всякое слабое ребро в направлении от A к B, а всякое сильное – в направлении от B к A. Полученный орграф обозначим через $\vec{G}$. Очевидно, что увеличивающая цепь в графе G существует тогда и только тогда, когда в графе $\vec{G}$ имеется ориентированный путь из какой-либо вершины x X в какую-либо вершину y Y. Такой путь можно найти (или убедиться в его отсутствии) простым поиском в ширину.

Задача об оптимальном каркасе (стягивающем дереве) состоит в следующем. Дан обыкновенный граф $G = (V, E)$ и весовая функция на множестве ребер $w: V \to \mathbf{R}$. Вес множества X E определяется как сумма весов составляющих его ребер. Требуется в графе G найти каркас максимального веса. Обычно рассматривают задачу на минимум, но это не существенно – она преобразуется в задачу на максимум, если заменить функцию w на $-w$. В этом разделе будем предполагать, что граф G связен, так что решением задачи всегда будет дерево.

Для решения задачи об оптимальном каркасе известно несколько жадных алгоритмов. Рассмотрим один из них, известный под названием *алгоритм Прима*. В этом алгоритме на каждом шаге рассматривается частичное решение задачи, представляющее дерево. Вначале это дерево состоит из единственной вершины, в качестве этой вершины может быть выбрана любая вершина графа. Затем к дереву последовательно добавляются ребра и вершины, пока не получится остовное дерево, т.е. каркас. Для того, чтобы из текущего дерева при добавлении нового ребра опять получилось дерево, это новое ребро должно соединять вершину дерева с вершиной, еще не принадлежащей дереву. Такие ребра будем называть *подходящими* относительно рассматриваемого дерева. «Жадность» алгоритма состоит в том, что на каждом шаге из всех подходящих ребер выбирается ребро наименьшего веса. Это ребро вместе с одно новой вершиной добавляется к дереву. Если обозначить через U и F множества вершин и ребер строящегося дерева, а через W множество вершин, еще не вошедших в это дерево, то алгоритм Прима можно представить следующим образом.

Алгоритм 1. *Построение оптимального каркаса методом Прима*

1 $U := \{a\}$, где a – произвольная вершина графа
2 $F :=$
3 $W := V - \{a\}$
4 **while** W **do**

5 найти ребро наибольшего веса $e = (x, y)$ среди всех таких ребер, у которых $x \quad U$, $y \quad W$

6 $F := F \quad \{e\}$

7 $U := U \quad \{y\}$

8 $W := W - \{y\}$

Докажем, что алгоритм Прима на самом деле строит оптимальный каркас. Дерево F назовем *фрагментом*, если оно является подграфом графа G и существует такой оптимальный каркас T_0 графа G, что F является подграфом графа T_0.

Теорема 4. *Если F – фрагмент, e – подходящее ребро наибольшего веса относительно F, то $F \quad \{e\}$ – фрагмент.*

Доказательство. Пусть T_0 – оптимальный каркас, содержащий F в качестве подграфа. Если ребро e принадлежит T_0, то $F \quad \{e\}$ – подграф T_0 и, следовательно, фрагмент. Допустим, e не принадлежит T_0. Если добавить ребро e к дереву T_0, то образуется цикл. В этом цикле есть еще хотя бы одно подходящее ребро относительно F (никакой цикл, очевидно, не может содержать единственное подходящее ребро). Пусть e – такое ребро. Тогда подграф $T_0 = T_0 - e + e$, получающийся из T_0 удалением ребра e и добавлением ребра e, тоже будет деревом. Так как $w(e) \quad w(e)$, то $w(T_0) \quad w(T_0)$. Но T_0 – оптимальный каркас, следовательно, $w(T_0) = w(T_0)$ и T_0 – тоже оптимальный каркас. Но $F \quad \{e\}$ является подграфом графа T_0 и, следовательно, фрагментом.

Дерево, состоящее из единственной вершины, очевидно, является фрагментом. Из теоремы 4 следует, что если после некоторого количества шагов алгоритма Прима дерево F является фрагментом, то оно будет фрагментом и после следующего шага. Следовательно, и окончательное решение, полученное алгоритмом, будет фрагментом, т.е. оптимальным каркасом.

Список использованной литературы:

1. Алексеев В.Е., Таланов В.А. Графы. Модели вычислений. Алгоритмы.– Н.Новгород, ННГУ, 2005. 307 с.
2. Зыков А.А. Основы теории графов.– М.: Вузовская книга, 2004. 664 с.
3. Касьянов В.Н., Евстигнеев В.А. Графы в программировании: обработка, визуализация и применение.– СПб.: БХВ-Петербург, 2003. 1104 с.
4. Галкина В. А Дискретная математика. Комбинаторная оптимизация на графах - М.: Издательство "Гелиос АРВ", 2003. — 232 с.
5. Extremal problems on graphs, directed graphs and hypergraphs. T.D.Townsend School of Mathematics The University of Birmingham, August 2015, 232 P.