Лекция 4. Стратегии поиска.

4.1. Оценки успеха при поиске цели.

Оценки успеха, характеризующие эффективность поиска, могут быть самыми различными. Будем полагать, что оценка успеха является положительным числом или нулем и чем больше число, тем успешнее действует агент. Тогда самая очевидная оценка — это возможность достижения цели вообще. Эта оценка носит двузначный характер: например, она принимает некоторое максимальное значение, если цель достижима, и минимальное в противном случае.

Другая оценка может быть ценой пути, вычисляемой с помощью функции цены пути.

Третья оценка может быть ценой поиска, характеризующейся объемом необходимых для поиска времени и памяти. Наконец, еще одна оценка успеха может быть общей ценой, являющейся некоторой функцией от цены пути и цены поиска.

Цена поиска зависит от многих обстоятельств. Так, например, если агент сильно стеснен во времени, то цена поиска может быть пропорциональна времени, затрачиваемому на поиск. Общая цена может зависеть от пройденного расстояния и затраченного на поиск времени. Вычисление общей цены непростая задача, поскольку выбор пути, а следовательно, и пройденное расстояние могут определяться временем, затраченным на поиск. Чем меньше времени потрачено на поиск, тем дальше от оптимального может оказаться выбор пути. Зачастую в этих условиях приходится идти на некий компромисс, далекий от оптимального решения.

Стратегии поиска различаются последовательностью или порядком просмотра состояний или множеств состояний в пространстве всех состояний среды. Стратегии поиска в пространстве состояний обычно сравнивают по ряду критериев. Основными критериями являются следующие.

Полнота: стратегия является полной при условии, что она всегда обеспечивает нахождение решения (целевого состояния или состояний), если оно вообще существует в данном пространстве состояний.

Сложность по времени: время, необходимое для нахождения решения.

Сложность по памяти: объем памяти, необходимый для решения задачи.

Оптимальность: стратегию называют оптимальной при условии, что она обеспечивает нахождение решения, которое может не быть наилучшим, но известно, что оно принадлежит некоторому классу или подмножеству, все элементы которого обладают неким общим свойством (или свойствами), согласно которому мы их относим к оптимальным решениям.

Минимальность: стратегия является минимальной, если она гарантирует нахождение наилучшего решения. Минимальность, таким образом, является частным и более сильным случаем оптимальности.

Стратегии поиска разбивают на две большие группы: слепой поиск и направленный поиск. Различие между слепым и направленным поиском можно продемонстрировать на следующем простом примере. Представим себе, что агент приехал в Россию по туристической путевке с целью совершить путешествие по

золотому кольцу России. Его самолет сел в аэропорту Шереметьево.

Обратный вылет тоже из Шереметьева. Путешествие по золотому кольцу агент совершал на автобусе. Посетив все города золотого кольца, агент самостоятельно поехал в близлежащий город Иваново, где задержался и отстал от автобуса. В отличие, например, от Ярославля, Иваново не имеет прямой автострады, соединяющей его с Москвой. У агента есть обратный билет на самолет с датой вылета на следующий день, билет не может быть продлен, да к тому же обратных билетов вообще нет на ближайшие две недели. Срок визы также заканчивается через два дня. Отправляясь в поездку, агент ставил перед собой, помимо туристических, несколько других целей: улучшить знание русского языка, завязать полезные деловые контакты, написать несколько пейзажей с видами русской природы и т.п. Но, учитывая серьезность ситуации, главной его целью теперь стало вовремя добраться до аэропорта Шереметьево, не опоздав на свой самолет. После формулировки главной цели агент может принимать во внимание и другие факторы, влияющие на ее достижение.

Прежде всего, следует решить, что мы будем понимать под действиями и состояниями. Если бы вдруг нам пришло в голову в качестве действий выбрать такие, как «повернуть руль на 10 градусов» или «продвинуться вперед на 1 метр», то решение задачи на таком уровне детализации стало бы практически неразрешимой задачей. В нашем случае в качестве действий лучше всего выбрать переезд из одного города в другой, а в качестве состояний нахождение агента в том или ином населенном пункте. Целевым состоянием в этом случае будет нахождение агента в аэропорту Шереметьево (в Москве).

Поиск будет слепым, если агент не имеет никакой информации о дорогах, ведущих из Иванова в другие города, и будет перебирать все дороги подряд. И наоборот, его поиск станет направленным, если он ознакомится с картой, увидит, что Москва лежит на юго-западе от Иванова, и будет пытаться выбирать дороги, ведущие в юго-восточном или близком к нему направлении.

4. 2. Слепой поиск

4.2.1. Поиск в ширину

Одна из самых очевидных стратегий поиска называется поиском в ширину. Поиск начинается с корневой вершины, определяются все последователи корневой вершины, затем все последователи каждого из последователей корневой вершины, далее все последователи каждого из последователей, найденных на предыдущем шаге, и т.д. до тех пор, пока не будут найдены все вершины, соответствующие целевым состояниям. Согласно этой стратегии, вершины глубиной k ищутся после того, как будут найдены все вершины глубиной k-1. На рис. 4. 1 показана последовательность нахождения вершин при поиске в ширину и при числе последователей каждой вершины, равном 2.

При поиске в ширину сначала рассматриваются все пути, длина которых равна 1, затем длиной 2 и т.д. Очевидно, что поиск в ширину удовлетворяет критерию полноты и минимальности, поскольку в процессе этого поиска рассматриваются все пути, ведущие из начальной вершины (состояния) во все остальные.

Оценим время и память, затрачиваемые в процессе поиска в ширину. Будем полагать, что число последователей каждой вершины равно l. Тогда в начале

поиска в ширину мы имеем одну начальную вершину, затем l последователей, потом l^2 последователей и т.д. В общем случае при глубине дерева поиска, равной к, число вершин, изучаемых в процессе поиска, равно $1 + l + l^2 + l^3 + ... + l^k$. При конкретных значениях l и k по этой формуле можно вычислить максимальное число вершин дерева поиска, которое может быть получено в процессе поиска в ширину. Естественно, решение можно найти раньше, чем будут найдены все последователи.

Однако для некоторых задач эта величина может быть достигнута. Обычно вместо этой формулы употребляют ее обозначение в виде $O(l^k)$, которое называют экспоненциальной оценкой сложности.

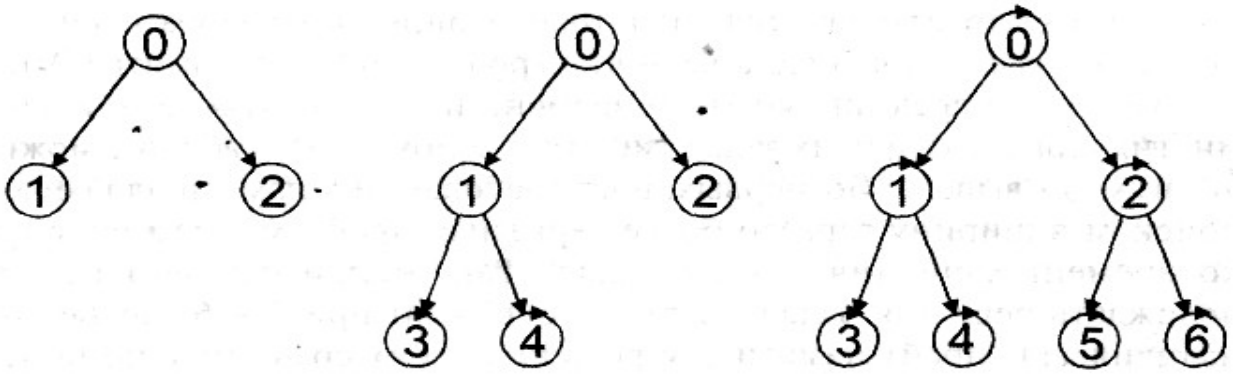


Рис. 4.1. Деревья поиска в ширину при l = 2: нулевая вершина (дерево глубиной k = 0), дерево после нахождения последователей нулевой вершины (дерево глубиной k = 1), дерево после нахождения последователей первой вершины, дерево после нахождения последователей второй вершины (дерево глубиной k = 2)

Если полагать, что получение каждого последователя требует одной единицы времени, то $O(l^k)$ является оценкой сложности по времени. В процессе поиска в ширину каждая вершина дерева должна сохраняться до получения требуемого решения. Если полагать, что для этого сохранения требуется единица памяти, то та же формула является одновременно и оценкой сложности по памяти для поиска в ширину.

Посмотрим, что это будут за величины при l=10 для различных значений k (табл. 4.1).

Предполагается, что 1000 вершин-последователей могут быть получены за 1 с, и что одна вершина требует 100 байт памяти. Эти значения соответствуют средним затратам времени и памяти, требуемым для решения многих иллюстративных задач типа головоломок. Из табл. 4.1 можно сделать два важных вывода.

Во-первых, рост требований к памяти для решения задач поиском в ширину гораздо более серьезная проблема, чем рост требований времени решения тех же задач.

Табл.4.1

Глубина дерева	Количество вершин	Требуемое для поиска время	Требуемая для поиска память

0	1	1 миллисекунда	100 байт
2	111	0,1 секунды	11 килобайт
4	11.111	11 секунд	1 мегабайт
6	10^6	18 минут	111 мегабайт
8	10^8	31 час	11 гигабайт
10	10^{10}	128 дней	1 терабайт
12	10^{12}	35 лет	111 терабайт
14	10^{14}	3500 лет	11111 терабайт

Так, например, вполне приемлемо подождать решения задачи в течение 18 мин при l = 6, но необходимость наличия 111 Мбайт памяти для решения таких сравнительно простых задач кажется слишком высокой ценой.

Во-вторых, для задач, в которых l > 10, помимо катастрофического увеличения требований к памяти, наблюдается стремительный рост временных затрат, не позволяющий решать реальные задачи с помощью поиска в ширину даже на современных мощных компьютерах. Это приводит к следующему заключению: Стратегии поиска, еющие экспоненциальные оценки сложности решения задач по времени и памяти, в частности стратегия поиска в ширину, нельзя использовать i решения задач большой размерности, т.е. задач, у которых l > 10.

4.2.2. Монотонный поиск в ширину

Поиск в ширину обеспечивает нахождение целевого состояния,

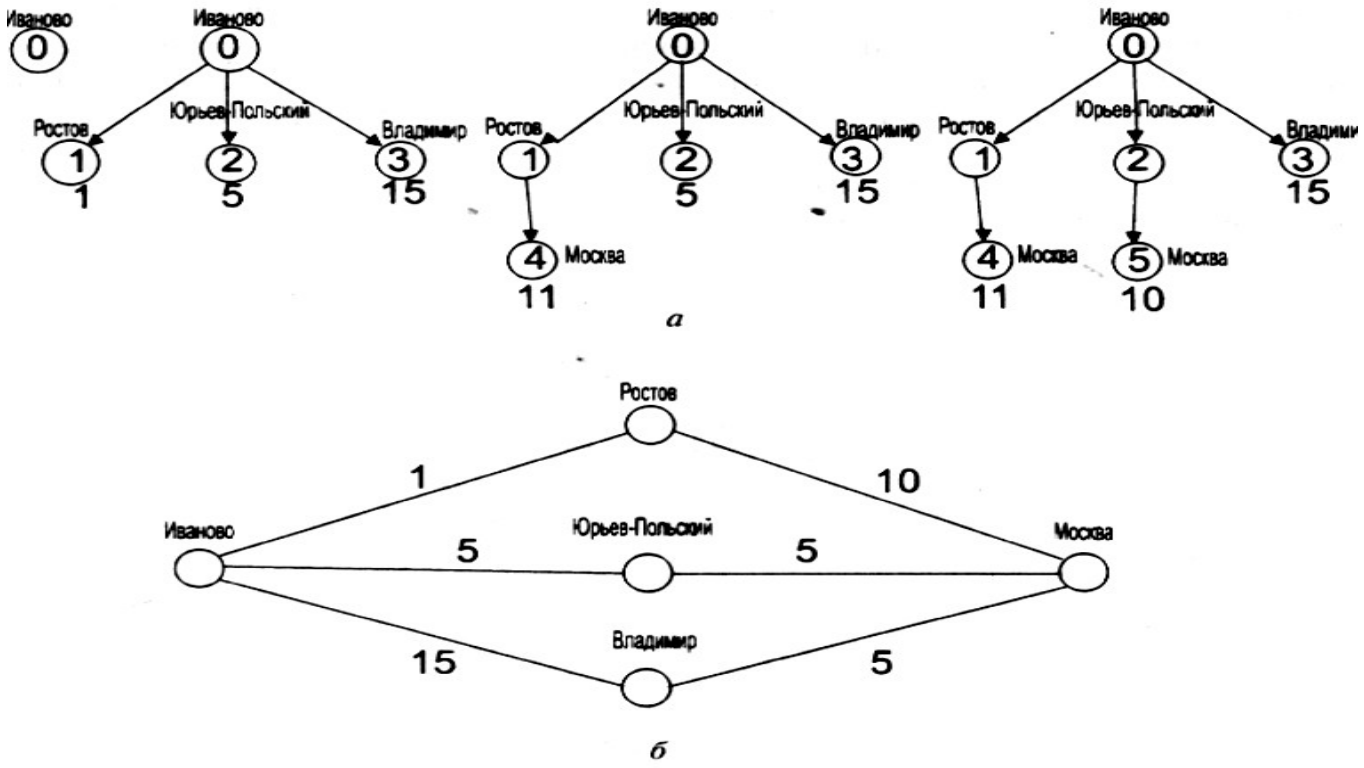
находящегося на минимальной глубине дерева поиска. Однако минимальность цены пути и этом не гарантируется. Монотонный поиск в ширину является некоторой модификацией поиска в ширину, заключающейся в том, что в процессе поиска в ширину всякий раз, когда определяется очередная вершина-последователь, одновременно вычисляется цена пути, ведущего в эту вершину. Эта цена пути используется для оценки целесообразности поиска вершин, продолжающих данный путь, сравнением с другими, уже полученными ценами путей.

Если цена данного пути меньше цен всех этих путей, то поиск вершин на этом пути продолжается. В противном случае поиск вершин на этом пути откладывается, а продолжается поиск вершин на путях, цена которых меньше данного. Когда на каком-либо пути будет достигнута целевая вершина, поиск на путях, цена которых выше цены этого пути, вообще прекращается. Все пути, имеющие одинаковую минимальную цену путей, ведущих в целевую вершину, являются решением задачи.

Вернемся к задаче поиска пути из Иванова в Москву. На рис. 4. 2, б схематически показана карта участка дорог, ведущих из Иванова в Москву. Каждая из дорог, ведущих из одного пункта в другой, снабжена числовой оценкой действия. На рис. 4.2, а показана последовательность монотонного поиска в ширину. Каждая из вершин помечена ценой пути, ведущего в эту вершину.

4.2.3. Поиск в глубину

При поиске в глубину, начиная с корневой вершины (корневая вершина



находится на уровне 1), рассматриваются все инцидентные ей вершины уровня 2, начиная слева направо. Если удается найти среди них все целевые вершины, то на этом поиск прекращается. Если среди них не удается найти все целевые вершины, а максимальная глубина дерева еще не достигнута, то берется самая левая вершина уровня 2 и рассматриваются все инцидентные ей вершины уровня 3 слева направо. Если после этого все целевые вершины все еще не найдены, то берется самая левая вершина из уровня 3. Затем снова рассматриваются все инцидентные ей вершины уровня 3 слева направо, и так до тех пор, пока либо не будут найдены все целевые вершины, либо достигнута максимальная глубина дерева, на которой в соответствии с рассматриваемой процедурой просмотрены все вершины, инцидентные самой левой вершине предыдущего уровня, и все целевые все еще не найдены. В последнем случае осуществляется подъем по дереву на один уровень вверх, выбор на этом уровне самой левой вершины, инцидентные которой вершины следующего уровня еще не рассмотрены, и поиск дальше среди целевых вершин по тому же принципу. И так до тех пор, пока все вершины дерева не будут рассмотрены.

Рис. 4. 2. Задача нахождения маршрута из Иванова в Москву:

а - порядок монотонного поиска в ширину. Каждая вершина помечена ценой пути, ведущим в нее из начальной вершины (снаружи вершины) и номером ее получения в процессе поиска (внутри вершины). На последнем шаге монотонного поиска в ширину найдена целевая вершина с ценой пути, равным 10;

б- пространство состояний, иллюстрирующее цену действий, соответствующую участкам пути между городами.

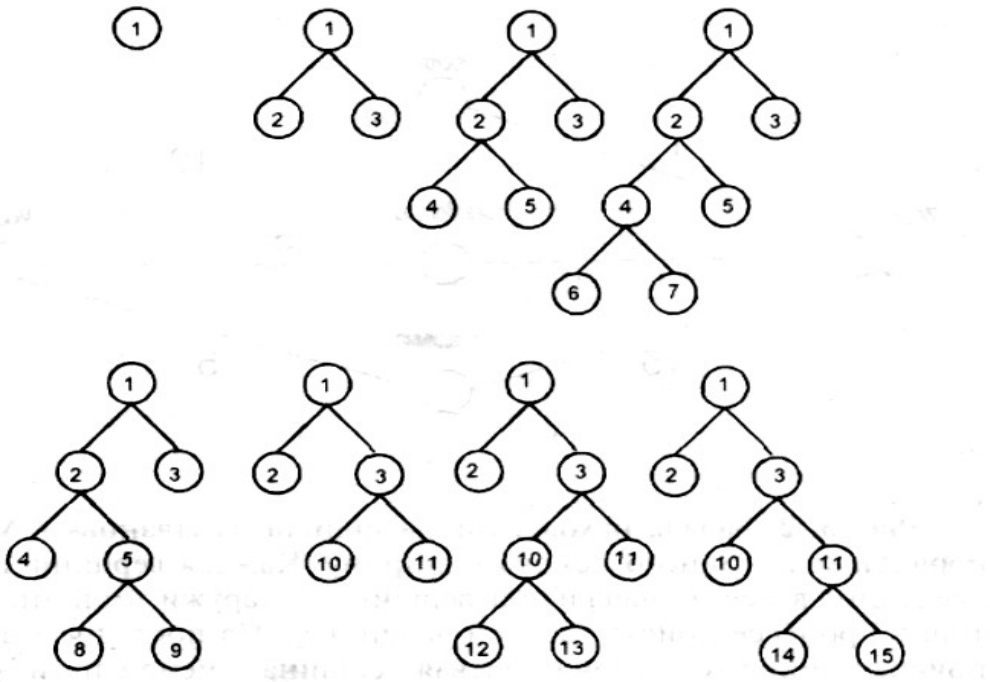


Рис. 4.3. Стратегия поиска в глубину

На рис. 4.3 показана последовательность поиска в глубину для дерева с тремя уровнями и степенью ветвления 2. Вершины на этом рисунке пронумерованы в соответствии с последовательностью их просмотра.

Требования к объему памяти при поиске в глубину существенно меньше, чем при поиске в ширину. Как видно из рис. 4. 3, в памяти необходимо хранить все вершины, инцидентные вершинам на пути из корневой вершины к любой другой, соседние вершины которой рассматриваются. Очевидно, что при степени ветвления l и глубине дерева k оценка сложности по памяти при поиске в глубину равна O(lk). Оценка сложности по времени для поиска в глубину остается такой же, как и при поиске в ширину, а именно $O(l^k)$.

Для задач, дерево поиска для которых конечно, а число целей сравнительно невелико, поиск в глубину может оказаться эффективным, так как имеет шанс найти это небольшое число целей без просмотра всех вершин.

Для задач, имеющих большую или даже бесконечную глубину дерева, поиск в глубину может оказаться крайне неэффективным, так как будет стремиться все время в глубину, в то время как цель может оказаться близкой к корню дерева, но на том пути, который еще не был просмотрен. Поиск же все время уходит вниз и в случае бесконечной глубины дерева может никогда не вернуться назад к просмотру вершин, среди которых находятся целевые. Вследствие этого следует избегать использования поиска в глубину для решения задач с большой или бесконечной глубиной дерева. Поиск в глубину, таким образом, нельзя считать ни полным, ни оптимальным.

4.2.4. Ограниченный поиск в глубину

Ограниченный поиск в глубину является частным случаем поиска в глубину.

При этом поиске используется ограничение на глубину поиска. Например, при поиске маршрута из Иванова в Москву при наличии информации о числе населенных пунктов, которые могут встретиться на этом пути, глубину поиска можно ограничить числом этих населенных пунктов. При ограниченном поиске в глубину, как только достигается уровень, совпадающий с этим числом, происходит возврат назад так же, как это делается при обычном поиске в глубину. Ограниченный поиск в глубину полный, если ограничение на глубину поиска выбрано таким образом, чтобы обеспечить просмотр всех вершин дерева. Оценки сложности по времени и памяти при ограниченном поиске в глубину те же, что и при обычном поиске в глубину.

4.2.5. Итеративный поиск в глубину

При ограниченном поиске в глубину сложность поиска зависит от выбора ограничения. Например, при поиске маршрута можно использовать не число всех населенных пунктов в районе поиска подходящего маршрута, а максимальное число населенных пунктов, которые могут встретиться на каждом из возможных маршрутов. Это число называют обычно радиусом поиска. Знание радиуса поиска приводит к более эффективному ограниченному поиску в глубину. Однако в большинстве задач радиус поиска не известен до тех пор, пока задача не будет решена. Итеративный поиск в глубину использует стратегию, основанную на итеративном применении ограниченного поиска в глубину сначала для радиуса поиска, равного 0, затем 2, после этого 3 и т.д. Четыре итерации такого поиска для бинарного дерева иллюстрирует рис. 4. 4. Итеративный поиск в глубину может показаться очень неэффективным, поскольку некоторые вершины просматриваются многократно. Однако для многих задач подавляющая доля вершин находится на низком уровне. Напомним,

что оценка сложности по времени при ограниченном поиске в глубину вычисляется по формуле

$$0(l^k) = 1 + l + l^2 + \ldots + l^{k-2} + l^{k-1} + l^k$$

Например, для l = 10, k = 5 будем иметь $0(1^k) = 111111$. При итеративном поиске на глубину k вершины глубины k просматриваются один раз, глубины k-1 — два раза, глубины k-2 — три и т.д. Корневая вершина просматривается k + 1 раз.

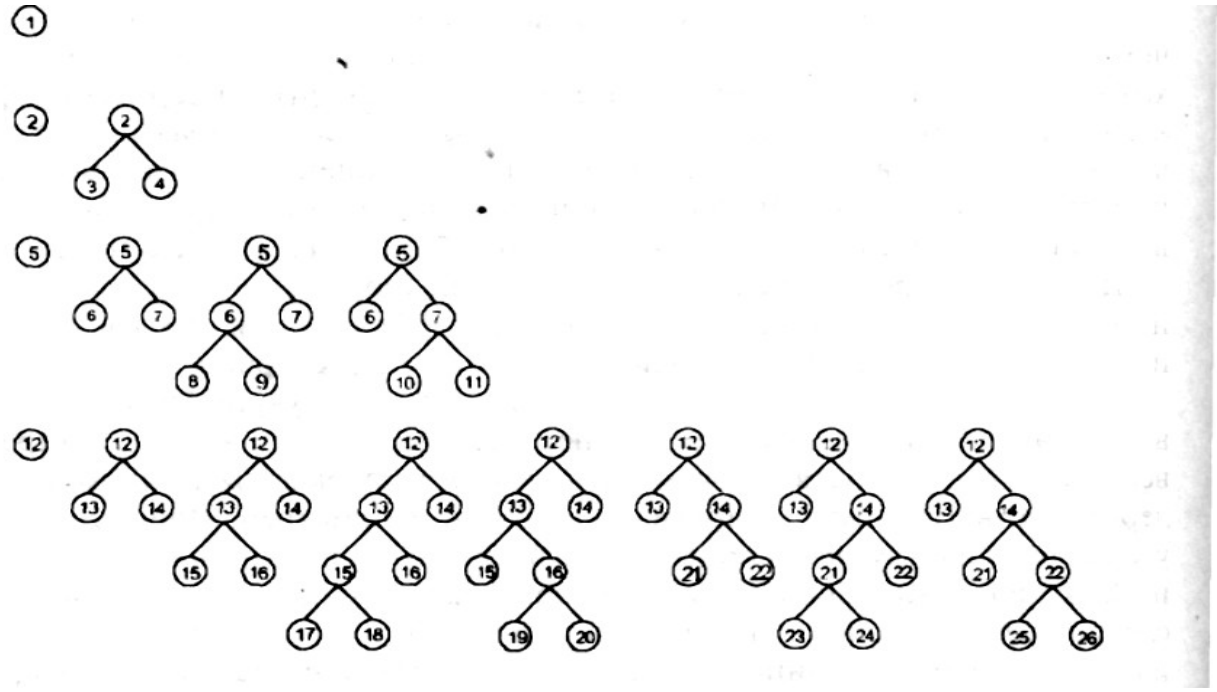


Рис. 4.4.Итеративный поиск в глубину.

Таким образом, оценка сложности по времени при итеративном поиске в глубину вычисляется по формуле

$$0(l^k) = (k+1)l^0 + (k)l^1 + (k-1)l^2 + ... + 3l^{k-2} + 2l^{k-1} + 1l^k$$

Для l = 10, k = 5 будем иметь $0(1^k)$ = 123456. По сравнению с оценкой сложности по времени для ограниченного поиска в глубину сложность по времени для итеративного поиска в глубину возросла только на 11%. Из формулы, приведенной выше, видно, что чем выше степень ветвления, тем ниже доля сложности, получающейся за счет повторного просмотра вершин. Но даже для случая, когда l = 2, сложность по времени итеративного поиска в глубину только в два раза превосходит сложность по времени простого поиска в глубину. Это означает, что оценка сложности по времени

итерационного поиска в глубину по-прежнему равна $O(l^k)$, а сложность по памяти - O(lk). Итеративный поиск в глубину достаточно хорошо себя зарекомендовал для задач с большим пространством поиска и неизвестной его глубиной.

Если цена данного пути меньше цен всех этих путей, то поиск вершин на этом пути продолжается. В противном случае поиск вершин на этом пути откладывается, а продолжается поиск вершин на путях, цена которых меньше данного. Когда на каком-либо пути будет достигнута целевая вершина, поиск на путях, цена которых

4.2.6. Двунаправленный поиск

Идея двунаправленного поиска основана на использовании сразу двух стратегий - прямого поиска от корневой вершины и обратного от целевой вершины. Процесс поиска прекращается, когда оба этих процесса встретятся на середине глубины. Если считать, что степень ветвления при прямом и обратном поиске одинакова, то оценка сложности по времени двунаправленного поиска будет $O(2l^{k/2}) = =O(l^{k/2})$. Эта оценка существенно лучше, чем аналогичная для

рассмотренных стратегий поиска. Однако все это в идеале, т.е. при условии, что цель только одна, степень ветвления и сложность нахождения последователей и предшественников одна и та же. При решении практических задач, однако, это условие может нарушаться. Кроме этого могут возникать и другие проблемы. Например, установление факта появления одной и той же вершины в той и другой половине поиска может потребовать значительных усилий и составить значительную долю сложности. Если все эти проблемы можно решить за постоянное время, не зависящее от числа вершин, то оценка сложности по времени двунаправленного поиска останется равной $0(l^{k/2})$.

4.2.7. Сравнение стратегий поиска

Характеристики шести рассмотренных стратегий, где l - степень ветвления, k — глубина поиска, d— максимальная глубина дерева поиска, r — ограничение на глубину поиска, сведены в табл. 4.2.

Таблица 4.2

критерии	Поиск					
	В ширину	Монотонный в ширину	В глубину	Ограниченный в глубину	Итеративный в глубину	Двунаправленный
время	l^k	l^k	l^d	l^r	l^k	$l^{k/2}$
память	l^k	l^k	l^d	l^r	l^k	$l^{k/2}$
Оптимальность	Да	Да	нет	Нет	да	Да
полнота	да	да	нет	Да,если $r \geq k$	да	да

4. 3. Направленный поиск

Как было показано в предыдущем разделе, все стратегии слепого поиска обладают экспоненциальными оценками сложности по времени поиска, а некоторые и по памяти, и, вследствие этого, пригодны для решения сравнительно небольших задач. В стратегиях слепого поиска знания, используемые при выборе очередной вершины, определяют только общий порядок выбора и никак не учитывают качество выбора, например, по сложности поиска целевой вершины. В стратегиях направленного поиска знания, используемые для выбора очередной вершины, более глубокие и используют специальные функции, называемые критериями. Если для каждой вершины b, участвующей в поиске, критерий может быть вычислен, а множество всех вершин упорядочивается по этому критерию, то выбор очередной вершины производится в соответствии со значением этого критерия. Чем лучше значение критерия (например, выше или ниже), тем предпочтительнее выбор. Иными словами, из множества альтернативных вершин выбирается та, для которой критерий имеет наилучшее значение. Поэтому стратегии выбора подобного типа называются стратегиями выбора по наилучшему критерию. Критерии обычно выбираются таким образом, чтобы оптимизировать сложность поиска, найти оптимальное решение или достичь того и другого. В настоящем разделе рассмотрим некоторые стратегии направленного поиска, при котором используются дополнительные знания о среде, позволяющие

понизить сложность поиска. Рассмотрим, как осуществляется направленный поиск при решении оптимизационных задач.

4.3.1. Поиск по критерию близости к цели

Критерием близости к цели обычно является числовая функция h(b)>=0 , вычисляемая для вершины b и характеризующая близость этой вершины к целевой. При использовании критерия близости к цели, начиная с корневой вершины, просматриваются все соседние ей вершины, выбирается та, для которой h(b) минимальна, и все повторяется вновь для выбранной вершины.

И так до тех пор, пока не будет достигнута целевая вершина, для которой h(b) = 0, т.е. выбирается та вершина, которая ближе всего находится к цели. Вид критерия близости к цели зависит от среды (является проблемно зависимым). Чтобы получить достаточно полное представление об этом критерии, вернемся к задаче поиска маршрута из Иванова в Москву. Критерием выбора в этом случае будет минимальное расстояние по прямой (кратчайшее расстояние) от вершины (населенного пункта) до Москвы. Естественно, чтобы значение критерия могло быть вычислено, необходимо иметь карту или какой-либо другой источник информации, содержащий сведения о кратчайших расстояниях от населенных пунктов до Москвы. На рис. 4. 5 показана последовательность поиска по критерию близости к цели для примера с поиском маршрута из Иванова в Москву, использованного при рассмотрении монотонного поиска в ширину.

На первом шаге вычисляется кратчайшее расстояние от корневой вершины (Иваново) до Москвы (h = 230). На втором шаге просматриваются все вершины (города), в которые ведет дорога из Иванова и вычисляется кратчайшее расстояние h от этих городов до Москвы. По этому критерию ближайшим по прямой до цели (Москвы) оказывается Юрьев- Польский (h = 140). На третьем шаге просматриваются все вершины, в которые ведет дорога из Юрьева-Польского.

Для них снова вычисляется кратчайшее расстояние h до Москвы, кратчайшим оказывается расстояние от Киржача (h = 76). Наконец, на последнем шаге вновь просматриваются все вершины, в которые ведет дорога из Киржача, а также вычисляется кратчайшее расстояние до этих городов. Среди этих городов оказывается город с кратчайшим расстоянием h = 0, т.е. целевая вершина (Москва).

На этом поиск по критерию близости к цели завершается. Поиск по критерию близости к цели является поиском в глубину, но с выбором на каждом шаге единственной вершины, от которой начинается следующий шаг.

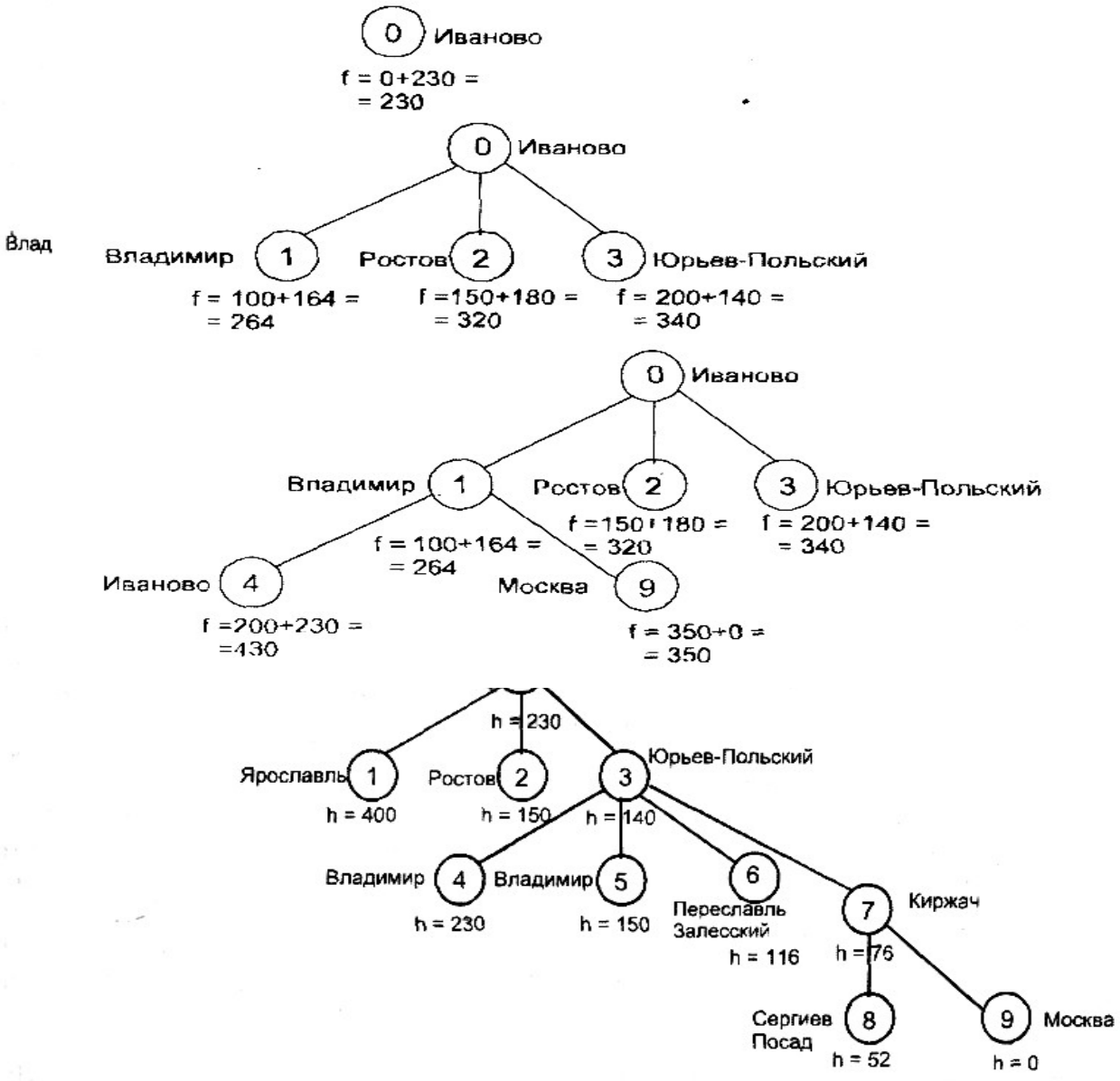


Рис. 4.5. Поиск по критерию близости к цели

Недостатки этой стратегии те же, что и для слепого поиска в глубину, а именно, она неоптимальная и неполная, поскольку может случиться, что поиск пойдет по бесконечному пути и никогда не вернется назад. Оценка сложности по времени этой стратегии поиска равна $0(l^d)$, где d— максимальная глубина пространства поиска. При поиске по критерию близости к цели приходится хранить все вершины, рассматриваемые при поиске. Поэтому оценка сложности по памяти для этой стратегии та же, что и оценка сложности по времени. Если критерий близости к цели выбран достаточно удачно, то сложность поиска может быть существенно уменьшена.

4.3.2. Поиск по критерию цены пути (A* - поиск)

С помощью этого критерия можно находить пути с минимальной ценой.

Обозначим g(b) - критерий цены пути из корневой вершины в вершину b, а h(b) - уже рассмотренный критерий близости к цели. Пусть оба критерия имеют одну и ту же размерность. Функцию f(b) = g(b) + h(b) можно считать критерием цены пути из корневой вершины в целевую.

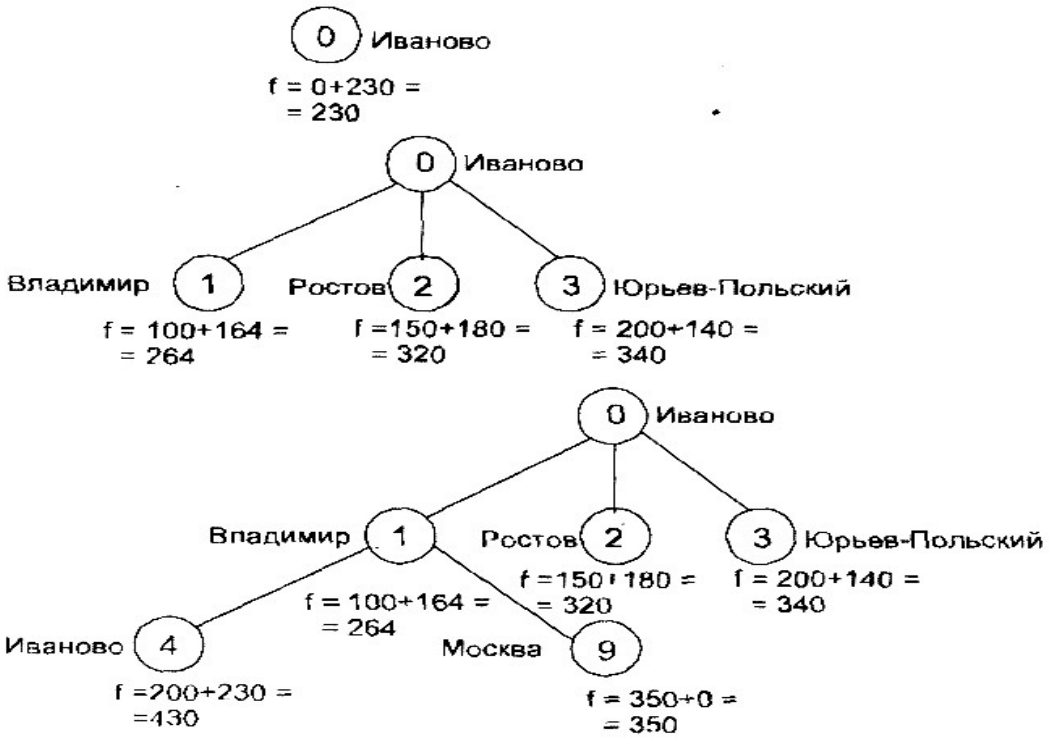


Рис.4.6. Поиск по критерию цены.

Рассмотрим стратегию поиска на основе этого критерия и покажем, что она будет полна и оптимальна, если ввести небольшие ограничения на критерий h(b).

Идея этого ограничения состоит в том, чтобы выбирать критерий h(b) таким образом, чтобы не переоценивать близость к цели, т.е. не выбирать значение h(b) меньше, чем оно есть на самом деле. Критерий h(b), выбираемый таким образом, называется допустимым. Стратегия поиска, использующая критерий f(b) и допустимый критерий h(b), известна как A* - поиск. Критерий близости цели h(b), использованный в примере с поиском маршрута из Иванова в Москву, является типичным примером допустимого критерия, поскольку не может быть пути из одного населенного пункта в другой короче, чем кратчайший путь. На рис. 4. 6 показаны первые четыре шага поиска пути из Иванова в Москву с использованием критерия f(b) в A*- поиске. Заметим, что в результате этого поиска, в отличие от поиска только по критерию близости к цели, рассмотренного в предыдущем разделе, выбран путь к Москве не через Юрьев- Польский, а через Владимир, хотя Юрьев- Польский ближе к Москве, чем Владимир. Такой выбор объясняется тем, что цена пути g(b) от Иванова до Юрьева- Польского выше, чем до Владимира вследствие очень плохой дороги. Дальнейший выбор маршрута можно проследить по рисунку, где на любом пути от корневой вершины значение критерия f(b) нигде не уменьшается при переходе к рассмотрению очередной вершины. Это не случайность и справедливо почти для всех допустимых критериев h(b) близости к цели. Критерии f(b), для которых имеет место подобное свойство, называются монотонными. Если монотонность нарушается, то путем незначительной коррекции это нарушение может быть устранено. Рассмотрим, например, пару вершин b, b', где b предшественник, а b' — последователь. Предположим, что

g(b) = 3, h(b) = 4. Тогда f(b) = 7, т.е. цена пути через вершину b самое меньшее равна 7. Предположим также, что g(b') = 4, h(b')= 2, т.е. f(b') = 6 . Ясно, что в этом случае критерий f(b) не является монотонным. К счастью, благодаря тому, что любой путь через b' является также путем через b, цена пути f(b') = 6 является бессмысленной. Поэтому каждый раз, как рассматривается какая-либо вершина b' и оказывается, что ее цена пути f(b') < f(b), мы полагаем, что f(b') =f(b), т.е. f(b') = max (f(b), f(b')).

Таким способом немонотонность может быть всегда устранена, и критерий f(b) никогда не будет уменьшаться вдоль одного и того же пути при условии, что h(b) допустимое. Если же f(b) никогда не уменьшается вдоль одного и того же пути, начинающегося от корневой вершины, то на пространство состояний можно наложить контуры, каждый из которых охватывает множество вершин b, для которых значение критерия f(b) меньше определенной величины c. Процесс поиска можно представить как переход от просмотра еще не просмотренных вершин какого-либо внутреннего контура, для всех вершин b_1 которого имеет место $f(b_1) < c$, к просмотру вершин некоторого внешнего контура, для всех вершин b_2 которого имеет место $f(b_2) < c_2$ и $c_1 < c_2$. Это продолжается до тех пор, пока внутри очередного контура не окажется целевая вершина с h(b) = 0. При удачном выборе критерия f(b) контуры как бы растягиваются в сторону целевой вершины, фокусируясь вокруг оптимального пути. Обозначим c* цену оптимального пути. Тогда можно утверждать следующее:

A*- поиск просматривает вершины с f(b) <=c*.

A*- поиск осуществляет направленный просмотр вершин, стремясь к просмотру вершин контура, для которых имеет место f(b) = c*.

Решение, получаемое с помощью A*- поиска, является оптимальным, поскольку находится вершина с максимальным значением f(b), а следовательно, и максимальным g(b), так как h(b) = 0. A*- поиск является также полным, поскольку, увеличивая постепенно значение критерия f(b), мы должны, в конце концов, найти путь к целевой вершине. Заметим также, что для данного критерия f(b) не существует никакой другой процедуры поиска, которая давала бы более оптимальные решения, чем A*- поиск. Все эти утверждения требуют более строгих доказательств, которые мы здесь не приводим. Оценки сложности по времени и памяти для A*- поиска аналогичны оценкам для поиска по критерию близости цели.

На идее A*- поиска построено много других методов поиска, учитывающих особенности конкретной среды, которые влияют на выбор критериев, и различные ограничения, например по доступным ресурсам (времени выполнения и доступной памяти).

4.3.3. Оптимизирующий итеративный поиск

Во многих задачах поиск целевого состояния (состояний) в пространстве состояний ставится как задача нахождения такого состояния (состояний), которое в определенном смысле наилучшее (оптимальное). При этом не имеет особого значения цена пути нахождения цели, если эта цель может быть достигнута за приемлемые затраты времени и памяти. Идея оптимизирующего итеративного поиска станет понятной, если полагать, что состояния — это точки на поверхности некоторого горного ландшафта. Чем выше точка находится над

уровнем моря, тем лучше (оптимальнее) состояние, которое она представляет. Точки, соответствующие пикам, являются оптимальными точками. Задачей оптимизирующего итеративного поиска является такой порядок просмотра точек, или иначе такое движение по поверхности ландшафта, при котором в конце концов достигаются (находятся) оптимальные точки. Имеется большое разнообразие процедур оптимизирующего итеративного поиска. Одной из таких процедур является градиентный поиск. Рассмотрим суть этой процедуры.

Идея градиентного поиска чрезвычайно проста - двигаться в направлении подъема вверх, начиная с некоторого начального состояния. Дерево поиска не хранится, а сохраняется только последнее найденное состояние b и оценка его качества L(b). Если попадаются несколько состояний с одинаковой оценкой качества, то, как правило, выбирается одно из них. Градиентный поиск в чистом виде обладает тремя известными недостатками.

Если вершин несколько, то поднявшись на одну из них, процесс поиска остановится, полагая, что цель достигнута, хотя достигнутая вершина является всего лишь локальным оптимумом и далека от наилучшей.

Если ландшафт имеет плато, то процесс поиска по нему становится случайным.

Если ландшафт имеет гребни (хребты) со слабым наклоном, то легко достигнув гребня, процесс поиска может очень долго идти по гребню, пока не достигнет оптимальной вершины.

После остановки процесса градиентного поиска начинается следующая итерация с другого начального состояния. Найденное оптимальное состояние сохраняется до тех пор, пока не будет найдено лучшее. Этот итеративный процесс поиска осуществляется конечное число раз или пока не будет найдено устраивающее решение.

Ясно, что если провести достаточное число подобных итераций, то оптимальное решение, в конце концов, будет найдено. Успех градиентного поиска сильно зависит от вида пространства состояний. Если число локальных минимумов невелико, то оптимальное решение будет найдено сравнительно быстро. Процед уры градиентного поиска могут отличаться способом выбора очередной вершины в процессе подъема на вершину, выбором очередной вершины для новой итерации и т.д.

Вопросы для самопроверки к главе 4:

1. Какие оценки успеха при поиске цели Вы знаете?
2. При слепом поиске в ширину исследуются сначала все состояния одного уровня, затем все состояния следующего уровня и т.д. или сначала все состояния одного направления, затем все состояния другого направления и т.д?
3. Является ли итеративный поиск одним из видов ограниченного поиска?
4. Что проще: поиск по критерию цены или поиск по критерию близости?
5. В чем различие между методом итерации и рекуррентными методами?