

Рис. 25.7. Метод локализации Монте-Карло, основанный на применении алгоритма фильтрации частиц для локализации мобильного робота: первоначальное состояние, глобальная неопределенность (а); приблизительно бимодальное состояние неопределенности после прохождения по (симметричному) коридору (б); унимодальное состояние неопределенности после перехода в офис, отличный от других (в)

На рис. 25.8 показано применение понятия линеаризации для (одномерной) модели движения робота. В левой части показана нелинейная модель движения $f(\mathbf{x}_t, \mathbf{a}_t)$ (параметр \mathbf{a}_t на этом графике не показан, поскольку он не играет никакой роли в этой линеаризации). В правой части эта функция аппроксимируется линейной функцией $f(\mathbf{x}_t, \mathbf{a}_t)$. График этой линейной функции проходит по касательной к кривой f в точке μ_t , которая определяет среднюю оценку состояния во время t . Такая линеаризация называется **разложением в ряд Тейлора** (первой степени). Фильтр Калмана, линеаризующий функции f и h с помощью разложения в ряд Тейлора, называется **расширенным фильтром Калмана** (или Extended Kalman Filter — EKF). На рис. 25.9 показана последовательность оценок, полученных роботом, который действует под управлением алгоритма локализации на основе расширенного фильтра Калмана. По мере передвижения робота неопределенность в оценке его местонахождения возрастает, как показано с помощью эллипсов погрешностей. Но как только робот начинает получать данные о дальности и азимуте до отметки с известным местонахождением, его погрешность уменьшается. Наконец, погрешность снова возрастает, как только робот теряет отметку из виду. Алгоритмы EKF действуют успешно, если отметки являются легко идентифицируемыми. Еще один вариант состоит в том, что распределение апостериорных вероятностей может быть мультимодальным, как показано на рис. 25.7, б. Задача, для решения которой требуется знать идентификацию отметок, представляет собой пример задачи **ассоциации данных**, которая обсуждалась в конце главы 15.

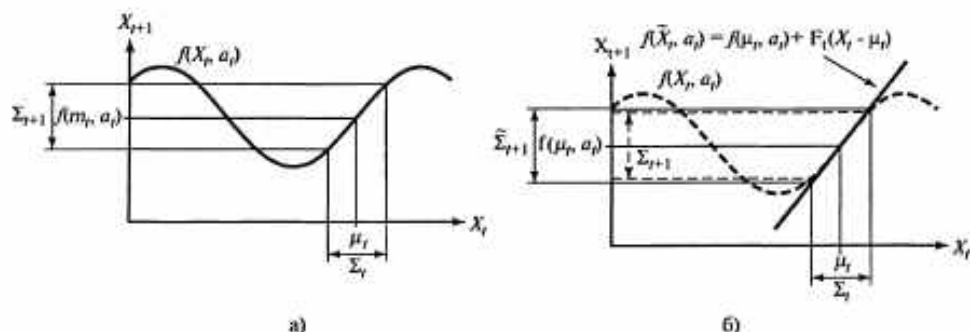


Рис. 25.8. Одномерная иллюстрация линеаризованной модели движения: функция f , проекция среднего μ_t и интервал ковариации (основанный на Σ_t) во время $t+1$ (а); линеаризованная версия представляет собой касательную к кривой функции f при значении μ_t . Проекция среднего μ_t определена правильно. Однако проекция ковариации Σ_{t+1} отличается от Σ_{t+1} (б)

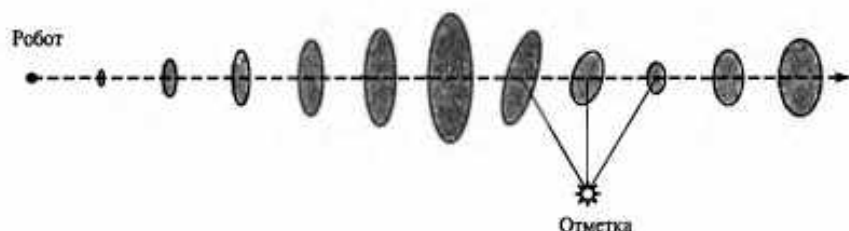


Рис. 25.9. Пример локализации с использованием расширенного фильтра Калмана. Робот движется по прямой. По мере его продвижения неопределенность постепенно возрастает, как показано с помощью эллипсов погрешностей. А после обнаружения роботом отметки с известной позицией неопределенность уменьшается

Составление карты

До сих пор в этой главе рассматривалась задача локализации одного объекта. Но в робототехнике поиск часто осуществляется в целях локализации сразу нескольких объектов. Классическим примером такой задачи является составление карты с помощью робота. Представьте себе робота, которому не дана карта его среды. Вместо этого он вынужден составлять такую карту самостоятельно. Безусловно, человечество добилося потрясающих успехов в искусстве составления карт, описывающих даже такие крупные объекты, как вся наша планета. Поэтому одна из задач, присущих робототехнике, состоит в создании алгоритмов, позволяющих роботам решать аналогичную задачу.

В литературе задачу составления карты роботом часто называют задачей **одновременной локализации и составления карты**, сокращенно обозначая ее как **SLAM** (Simultaneous Localization And Mapping). Робот не только обязан составить карту, но и должен сделать это, изначально не зная, где он находится. SLAM — одна из наиболее важных задач в робототехнике. Мы рассмотрим ту версию этой задачи, в которой среда является фиксированной. Даже этот более простой вариант задачи с большим трудом поддается решению; но положение становится значительно сложнее, когда в среде допускается возникновение изменений в ходе перемещения по ней робота.

С точки зрения статистического подхода задача составления карты сводится к задаче байесовского алгоритмического вывода, так же как и локализация. Если, как и прежде, карта будет обозначаться через M , а поза робота во время t — через \mathbf{X}_t , то можно переформулировать уравнение 25.1, чтобы включить данные обо всей карте в выражение для апостериорной вероятности:

$$\begin{aligned} & \mathbf{P}(\mathbf{X}_{t+1}, M | \mathbf{z}_{1:t+1}, \mathbf{a}_{1:t}) \\ &= \alpha \mathbf{P}(\mathbf{z}_{t+1} | \mathbf{X}_{t+1}, M) \int \mathbf{P}(\mathbf{X}_{t+1} | \mathbf{x}_t, \mathbf{a}_t) \mathbf{P}(\mathbf{x}_t, M | \mathbf{z}_{1:t}, \mathbf{a}_{1:t-1}) d\mathbf{x}_t \end{aligned}$$

На основании этого уравнения фактически можно сделать некоторые благоприятные для нас выводы: распределения условных вероятностей, необходимые для включения данных о действиях и измерениях, по существу являются такими же, как и в задаче локализации робота. Единственная предосторожность связана с тем, что новое пространство состояний (пространство всех поз робота и всех карт) имеет гораздо больше измерений. Достаточно представить себе, что принято решение изобразить конфигурацию всего здания с фотографической точностью. Для этого, по-видимому, потребуются сотни миллионов чисел. Каждое число будет представлять собой случайную переменную и вносить свой вклад в формирование чрезвычайно высокой размерности пространства состояний. Эта задача еще в большей степени усложняется в связи с тем фактом, что робот может даже не знать заранее о том, насколько велика его среда. Это означает, что ему придется динамически корректировать размерность M в процессе составления карты.

По-видимому, одним из наиболее широко применяемых методов решения задачи SLAM является EKF. Обычно этот метод используется в сочетании с моделью восприятия данных об отметках и требует, чтобы все отметки были различимыми. В предыдущем разделе апостериорная оценка была представлена с помощью гауссова распределения со средним μ_t и ковариацией Σ_t . При использовании для решения

задачи SLAM подхода, основанного на методе EKF, это распределение апостериорных вероятностей снова становится гауссовым, но теперь среднее μ_t выражается в виде вектора с гораздо большим количеством измерений. В нем представлена не только поза робота, но и местонахождение всех характеристик (или отметок) на карте. Если количество таких характеристик равно n , то вектор будет иметь размерность $2n+3$ (два значения требуются для указания местонахождения отметки и три — для указания позы робота). Следовательно, матрица Σ_t имеет размерность $(2n+3) \times (2n+3)$ и следующую структуру:

$$\Sigma_t = \begin{pmatrix} \Sigma_{xx} & \Sigma_{xm} \\ \Sigma_{xm}^T & \Sigma_{mm} \end{pmatrix} \quad (25.2)$$

В этом уравнении Σ_{xx} — ковариация данных о позе робота, которая уже рассматривалась в контексте локализации; Σ_{xm} — матрица с размерами $3 \times 2n$, которая выражает корреляцию между характеристиками на карте и координатами робота. Наконец, Σ_{mm} — это матрица с размерами $2n \times 2n$, которая задает ковариацию характеристик на карте, включая все парные корреляции. Поэтому потребность в памяти для алгоритмов, основанных на методе EKF, измеряется квадратичной зависимостью от n (количества характеристик на карте), а время обновления также определяется квадратичной зависимостью от n .

Прежде чем перейти к изучению математических выкладок, рассмотрим решение задачи по методу EKF на графиках. На рис. 25.10 показано, как робот движется в среде с восемью отметками, расположенными в два ряда по четыре отметки каждый. Первоначально робот не имеет информации о том, где находятся отметки. Предполагается, что каждая отметка имеет другой цвет, и робот может надежно отличать их друг от друга. Робот начинает двигаться влево, в заранее заданном направлении, но постепенно теряет уверенность в том, есть ли у него достоверная информация о своем местонахождении. Эта ситуация показана на рис. 25.10, а с помощью эллипсов погрешности, ширина которых возрастает по мере дальнейшего передвижения робота. Двигающийся робот получает данные о дальности и азимуте до ближайших отметок, а эти наблюдения используются для получения оценок местонахождения таких отметок. Естественно, что неопределенность в оценке местонахождения этих отметок тесно связана с неопределенностью локализации робота. На рис. 25.10, б, в показано изменение доверительного состояния робота по мере того, как он продвигается в своей среде все дальше и дальше.

Важной особенностью всех этих оценок (которую не так уж легко заметить, рассматривая приведенные графические изображения) является то, что в рассматриваемом алгоритме поддерживается единственное гауссово распределение по всем оценкам. Эллипсы погрешностей на рис. 25.10 представляют собой проекции этого гауссова распределения на подпространство координат робота и отметки. Эта многомерное гауссово распределение апостериорных вероятностей поддерживает корреляции между всеми оценками. Данное замечание приобретает важное значение при попытке понять, что происходит на рис. 25.10, г. На этом рисунке показано, что робот обнаруживает отметку, ранее нанесенную на карту. В результате его собственная неопределенность резко уменьшается. Такое же явление происходит и с неопределенностью всех других отметок. Указанное событие является следствием того факта, что оценка местонахождения робота и оценки местонахождений отметок имеют высокую степень корреляции в гауссовом распределении апостериорных вероятностей. Надежное выяв-

ление знаний об одной переменной (в данном случае о позе робота) автоматически приводит к уменьшению неопределенности всех других переменных.

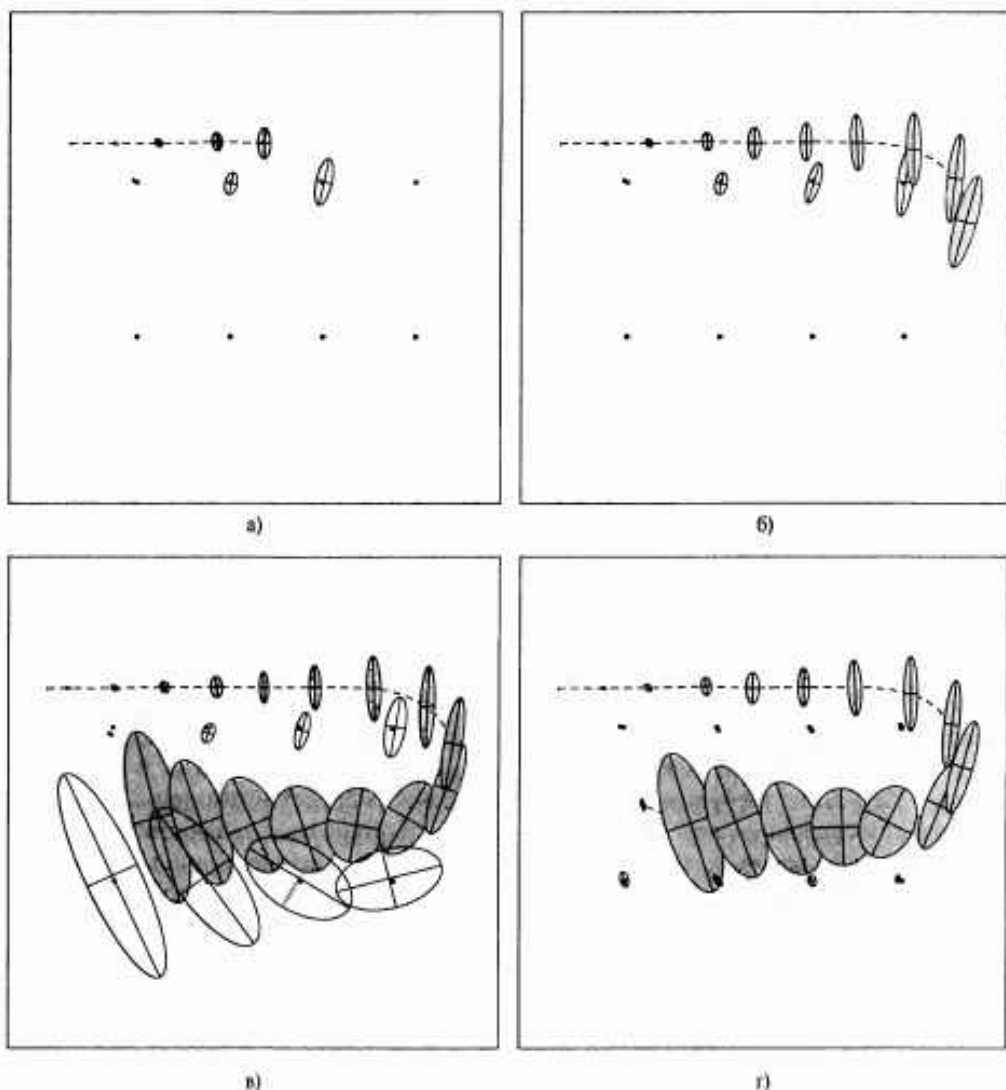


Рис. 25.10. Применение метода EKF для решения задачи составления карты роботом. Путь робота обозначен штриховой линией, а его оценки собственного положения — затененными эллипсами. Восемь различных отметок с неизвестными местонахождениями показаны в виде небольших точек, а оценки их местонахождения показаны в виде белых эллипсов: неопределенность робота в отношении его позиции возрастает, так же как и его неопределенность в отношении встреченных им отметок; этапы, на протяжении которых робот встречает новые отметки и наносит их на карту с возрастающей неопределенностью (а–в); робот снова встречает первую отметку, и неопределенность всех отметок уменьшается благодаря тому факту, что все эти оценки являются коррелированными (г)

Алгоритм ЕKF составления карты напоминает алгоритм локализации ЕKF, описанный в предыдущем разделе. Основное различие между ними определяется тем, что в распределении апостериорных вероятностей учитываются дополнительные переменные отметок. Модель движения для отметок является тривиальной, поскольку они не движутся. Таким образом, для этих переменных функция f представляет собой единичную функцию, а функция измерения по существу остается такой же, как и прежде. Единственное различие состоит в том, что якобиан H_t в уравнении обновления ЕKF берется не только по отношению к позе робота, но также и по отношению к местонахождению отметки, которая наблюдалась во время t . Результирующие уравнения ЕKF являются еще более устрашающими по своей сложности, чем те, которые были сформулированы перед этим; именно по этой причине мы их здесь опускаем.

Однако существует еще одна сложность, которая до сих пор нами игнорировалась, — тот факт, что размер карты M заранее не известен. Поэтому не известно также количество элементов в окончательной оценке μ_t и Σ_t . Эти данные приходится перепределять динамически, по мере обнаружения роботом все новых и новых отметок. Но эту проблему можно решить чрезвычайно просто — как только робот обнаруживает новую отметку, он добавляет новый элемент к распределению апостериорных вероятностей. Если же значение дисперсии этого нового элемента инициализируется очень большим числом, то результирующее распределение апостериорных вероятностей становится таким же, как если бы робот заранее знал о существовании этой отметки.

Другие типы восприятия

Но не все средства робототехнического восприятия предназначены для локализации и составления карт. Роботов наделяют также способностями воспринимать температуру, запахи, акустические сигналы и т.д. Многие из этих измеряемых величин могут подвергаться вероятностной оценке, как и при локализации и составлении карт. Для этого требуется лишь то, чтобы такими средствами оценки служили распределения условных вероятностей, которые характеризуют эволюцию переменных состояния во времени, а также другие распределения, которые описывают связь между результатами измерений и переменными состояния.

Однако не все практически применяемые в робототехнике системы восприятия опираются на вероятностные представления. Фактически, несмотря на то, что внутреннее состояние во всех рассматриваемых выше примерах имело четкую физическую интерпретацию, такая ситуация не обязательно наблюдается в действительности. Например, представьте себе шагающего робота, который пытается перенести ногу над препятствием. Допустим, этот робот действует согласно такому правилу, что он вначале должен поднять ногу на небольшую высоту, а затем поднимать ее все выше и выше, если предыдущее значение высоты не позволяет избежать столкновения ноги с препятствием. Можно ли утверждать, что указанная в команде на выполнение этого движения высота подъема ноги является представлением некоторой физической величины в реальном мире? Возможно, что эта высота действительно как-то связана с высотой и крутизной препятствия. Но в данном случае высоту подъема ноги можно также рассматривать как вспомогательную переменную в алгоритме работы контроллера робота, лишенную непосредственного физического смысла. Подобные способы представления нередко применяются в робототехнике и вполне подходят для решения определенных задач.

В настоящее время в робототехнике ясно выражена тенденция к использованию представлений с полностью определенной семантикой. Но вероятностные методы превосходят другие подходы по своей производительности в решении многих трудных задач восприятия, таких как локализация и составление карт. Тем не менее статистические методы иногда становятся слишком громоздкими, поэтому на практике могут оказаться столь же эффективными более простые решения. Самым лучшим учителем, позволяющим понять, какого подхода действительно следует придерживаться, является опыт работы с реальными физическими роботами.

25.4. ПЛАНИРОВАНИЕ ДВИЖЕНИЙ

В робототехнике принятые решения в конечном итоге воплощаются в движениях исполнительных механизмов. Задача **позиционирующего движения** состоит в доставке робота или его конечного исполнительного механизма в заданную целевую позицию. Это — сложная задача, но еще сложнее задача **согласующего движения**, при выполнении которой робот движется, находясь в физическом контакте с препятствием. Примером согласующего движения является закручивание электрической лампочки манипулятором робота или подталкивание роботом ящика для его перемещения по поверхности стола.

Начнем с поиска подходящего представления, которое позволяло бы описывать и решать задачи планирования движений. Как оказалось, более удобным для работы по сравнению с исходным трехмерным пространством является **пространство конфигураций** — пространство состояний робота, определяемых положением, ориентацией и углами поворота шарниров. Задача **планирования пути** состоит в поиске пути от одной конфигурации к другой в пространстве конфигураций. В этой книге уже встречались различные версии задачи планирования пути, а в робототехнике основной характерной особенностью планирования пути является то, что в этой задаче должны рассматриваться непрерывные пространства. В литературе по робототехническому планированию пути рассматривается широкий набор различных методов, специально предназначенных для поиска путей в непрерывных пространствах с большим количеством измерений. Основные семейства применяемых при этом подходов известны под названиями **декомпозиции ячеек** и **скелетирования**. В каждом из этих подходов задача планирования непрерывного пути сводится к задаче поиска в дискретном графе на основе выявления некоторых канонических состояний и путей в свободном пространстве. Во всем данном разделе предполагается, что движения детерминированы, а информация о локализации робота является точной. В следующих разделах эти предположения будут ослаблены.

Пространство конфигураций

Первый шаг к решению задачи управления движением робота состоит в создании подходящего представления задачи. Начнем с простого представления для простой задачи. Рассмотрим манипулятор робота, показанный на рис. 25.11, а. В нем имеются два шарнира, которые движутся независимо друг от друга. В результате движения шарниров изменяются координаты (x, y) локтя и захвата (манипулятор не может двигаться в направлении z). Это описание показывает, что конфигурацию данного

робота можно описать с помощью четырехмерных координат: использовать координаты (x_e, y_e) для обозначения местонахождения локтя относительно среды и координаты (x_g, y_g) — для обозначения местонахождения захвата. Очевидно, что эти четыре координаты полностью характеризуют состояние робота. Они составляют представление, которое принято называть представлением **рабочего пространства**, поскольку координаты робота заданы в той же системе координат, что и объекты, которыми он должен манипулировать (или столкновения с которыми должен избегать). Представления рабочего пространства хорошо подходят для проверки на предмет столкновения, особенно если робот и все объекты представлены с помощью простых многоугольных моделей.

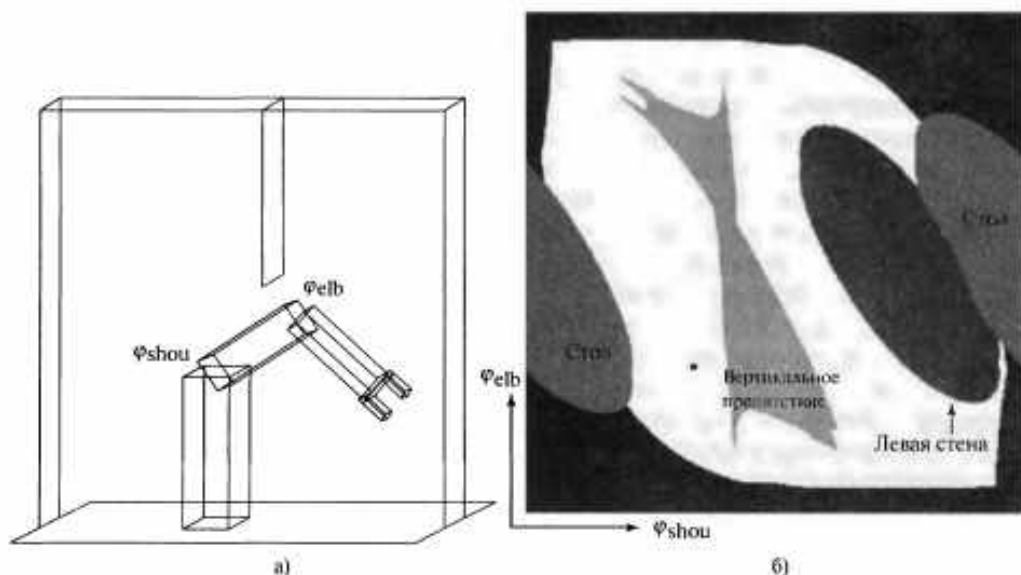


Рис. 25.11. Сравнение способов представления: представление рабочего пространства манипулятора робота с двумя степенями свободы; рабочее пространство представляет собой ящик с плоским препятствием в виде пластины, свисающей с потолка (а); пространство конфигураций того же робота. В этом пространстве только участки, обозначенные белым цветом, соответствуют конфигурациям, в которых не возникают столкновения с препятствиями. Точка на этом рисунке соответствует конфигурации робота, показанной слева (б)

Но представление рабочего пространства имеет один недостаток, связанный с тем, что фактически не все координаты рабочего пространства являются достижимыми, даже в отсутствии препятствий. Это обусловлено наличием **ограничений связи** в пространстве достижимых координат рабочего пространства. Например, позиция локтя (x_e, y_e) и позиция захвата (x_g, y_g) всегда разнесены на постоянное расстояние, поскольку шарниры, соответствующие этим позициям, связаны с помощью жесткого предплечья. Планировщик движений робота с алгоритмом, определенным на координатах рабочего пространства, сталкивается с проблемой выработки таких путей, которые позволяют придерживаться указанных ограничений. Такая задача становится особенно сложной в связи с тем, что пространство состояний является непрерывным, а ограничения — нелинейными.

Как оказалось, проще осуществлять планирование на основе представления **пространства конфигураций**. Вместо представления состояния робота с помощью декартовых координат его элементов это состояние представляется с помощью конфигурации шарниров робота. В данном примере в конструкцию робота входят два шарнира. Поэтому его состояние может быть представлено двумя углами, φ_5 и φ_6 , относящимися соответственно к шарниру плеча и к шарниру локтя. В отсутствие каких-либо препятствий робот может свободно выбрать любое значение из пространства конфигураций. В частности, при планировании пути можно связать текущую и целевую конфигурацию прямой линией. В таком случае, следуя по этому пути, робот может просто изменять углы поворота своих шарниров с постоянной скоростью до тех пор, пока не будет достигнуто целевое местонахождение.

К сожалению, и подход на основе пространства конфигураций имеет свои недостатки. Задание для робота обычно выражается в координатах рабочего пространства, а не в координатах пространства конфигураций. Например, может потребоваться, чтобы робот поместил свой конечный исполнительный механизм в определенную координату в рабочем пространстве, возможно, с указанием также его ориентации. В связи с этим возникает вопрос: как отобразить такие координаты рабочего пространства на пространство конфигураций? Вообще говоря, проще поддается решению обратная задача — преобразование координат пространства конфигураций в координаты рабочего пространства, поскольку для этого достаточно выполнить ряд совершенно очевидных преобразований координат. Эти преобразования являются линейными для призматических шарниров и тригонометрическими для поворотных шарниров. Такую цепочку преобразований координат принято называть **кинематикой**; этот термин уже встречался при обсуждении мобильных роботов.

Обратная задача вычисления конфигурации робота, для которого задано местонахождение исполнительного механизма в координатах рабочего пространства, называется **обратной кинематикой**. Проблема вычисления обратной кинематики, как правило, является трудной, особенно для роботов со многими степенями свободы. В частности, это решение редко является уникальным. Для рассматриваемого в качестве примера манипулятора робота существуют две различные конфигурации, при которых захват занимает одни и те же координаты рабочего пространства (см. рис. 25.11).

Вообще говоря, для любого множества координат рабочего пространства этого манипулятора робота с двумя сочленениями количество обратных кинематических решений изменяется в пределах от нуля до двух. А для большинства промышленных роботов количество решений бесконечно велико. Чтобы понять, почему это возможно, достаточно представить себе, что в роботе, рассматриваемом в качестве примера, будет установлен третий, дополнительный шарнир, ось вращения которого параллельна оси существующего шарнира. В таком случае можно будет поддерживать фиксированное местонахождение (но не ориентацию!) захвата и вместе с тем свободно вращать его внутренние шарниры в большинстве конфигураций робота. Добавив еще несколько шарниров (определите, сколько именно?), можно добиться того же эффекта, поддерживая также постоянную ориентацию. Пример аналогичной ситуации уже рассматривался в данной книге, когда читателю было предложено провести “эксперимент”, положив ладонь на стол и двигая локтем. В таком случае кинематическое ограничение на позицию ладони не позволяет однозначно определить конфигурацию локтя. Иными словами, задача определения обратной кинема-

тики для сочленения “плечо—предплечье” руки с ладонью, лежащей на столе, имеет бесконечное количество решений.

Вторая проблема, возникающая при использовании представлений пространства конфигураций, связана с наличием препятствий, которые могут существовать в рабочем пространстве робота. В примере, приведенном на рис. 25.11, *а*, показано несколько таких препятствий, включая свободно свисающую с потолка полосу, которая проникает в самый центр рабочего пространства робота. В рабочем пространстве такие препятствия рассматриваются как простые геометрические формы, особенно в большинстве учебников по робототехнике, которые в основном посвящены описанию многоугольных препятствий. Но как эти препятствия выглядят в пространстве конфигураций?

На рис. 25.11, *б* показано пространство конфигураций робота, рассматриваемого в качестве примера, при той конкретной конфигурации препятствий, которая приведена на рис. 25.11, *а*. Это пространство конфигураций можно подразделить на два подпространства: пространство всех конфигураций, достижимых для робота, которое принято называть **свободным пространством**, и пространство недостижимых конфигураций, называемое **занятым пространством**. Обозначенный белым цветом участок на рис. 25.11, *б* соответствует свободному пространству. Все другие участки соответствуют занятому пространству. Различные затенения в занятом пространстве соответствуют разным объектам в рабочем пространстве робота; участки, выделенные черным цветом и окружающие все свободное пространство, соответствуют конфигурациям, в которых робот сталкивается сам с собой. Можно легко обнаружить, что подобные нарушения в работе возникают при крайних значениях углов поворота шарниров плеча или локтя. Два участка овальной формы по обе стороны от робота соответствуют столу, на котором смонтирован робот. Аналогичным образом, третий овальный участок соответствует левой стене. Наконец, наиболее интересным объектом в пространстве конфигураций является простое вертикальное препятствие, проникающее в рабочее пространство робота. Этот объект имеет любопытную форму: он чрезвычайно нелинеен, а в некоторых местах является даже вогнутым. Приложив немного воображения, читатель легко узнает форму захвата на верхнем левом конце манипулятора. Рекомендуем читателю на минуту задержаться и изучить эту важную схему. Форма рассматриваемого препятствия не так уж очевидна! Точка внутри рис. 25.11, *б* обозначает конфигурацию робота, как показано на рис. 25.11, *а*. На рис. 25.12 изображены три дополнительные конфигурации как в рабочем пространстве, так и в пространстве конфигураций. В конфигурации “conf-1” захват окружает вертикальное препятствие.

Вообще говоря, даже если рабочее пространство робота представлено с помощью плоских многоугольников, форма свободного пространства может оказаться очень сложной. Поэтому на практике обычно применяется ощупывание пространства конфигураций вместо явного его построения. Планировщик может вырабатывать конфигурацию, а затем проверять ее для определения того, находится ли она в свободном пространстве, применяя кинематику робота и определяя наличие столкновений в различных координатах рабочего пространства.

Методы декомпозиции ячеек

В указанном выше первом подходе к планированию пути используется **декомпозиция ячеек**; иными словами, в этом методе осуществляется разложение

свободного пространства на конечное количество непрерывных участков, называемых *ячейками*. Эти участки обладают тем важным свойством, что задача планирования пути в пределах одного участка может быть решена с помощью простых средств (например, в виде передвижения по прямой линии). Таким образом, задача планирования пути преобразуется в задачу поиска в дискретном графе, во многом аналогичную задачам поиска, представленным в главе 3.

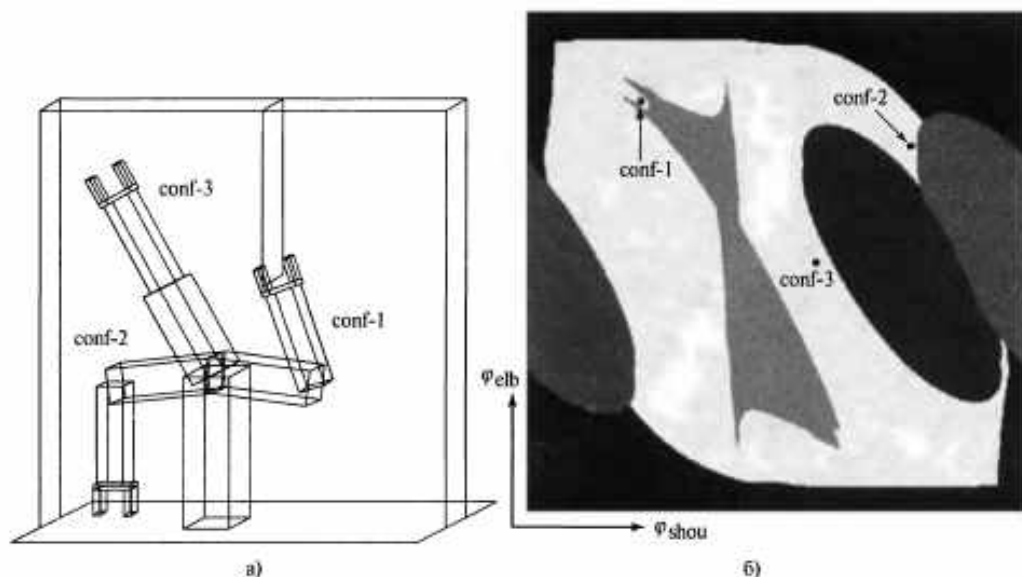


Рис. 25.12. Три конфигурации робота, показанные в рабочем пространстве и пространстве конфигураций

Простейшая декомпозиция ячеек представляет собой сетку с равномерным шагом. На рис. 25.13, *а* показаны декомпозиция пространства с помощью квадратной сетки и путь решения, оптимальный для сетки с этими размерами. Кроме того, на этом рисунке используются затенение в виде градаций серого цвета для обозначения стоимости каждой ячейки сетки свободного пространства, т.е. стоимости самого короткого пути от этой ячейки к цели. (Эти стоимости можно вычислить с помощью детерминированной формы алгоритма Value-Iteration, приведенного в листинге 17.1.) На рис. 25.13, *б* показана соответствующая траектория манипулятора в рабочем пространстве.

Такая декомпозиция имеет преимущество в том, что обеспечивает чрезвычайно простую реализацию, но характеризуется также двумя ограничениями. Во-первых, она может применяться только для пространств конфигураций с малым количеством измерений, поскольку количество ячеек сетки растет экспоненциально в зависимости от d , т.е. от количества измерений. Во-вторых, возникает проблема, обусловленная тем, что некоторые ячейки являются «смешанными», т.е. не принадлежащими полностью ни к свободному, ни к занятому пространству. Путь, найденный в качестве решения, который включает такую ячейку, может не соответствовать действительному решению, в связи с тем что не будет существовать способа пересечения ячейки в желаемом направлении по прямой линии. В результате этого процеду-

ра планирования пути становится противоречивой. С другой стороны, если мы будем настаивать на том, чтобы использовались только полностью свободные ячейки, то процедура планирования станет неполной, в связи с тем что могут возникать случаи, в которых единственные возможные пути к цели лежат через смешанные ячейки, особенно если размер ячейки сопоставим с размерами проходов и просветов в рассматриваемом пространстве.

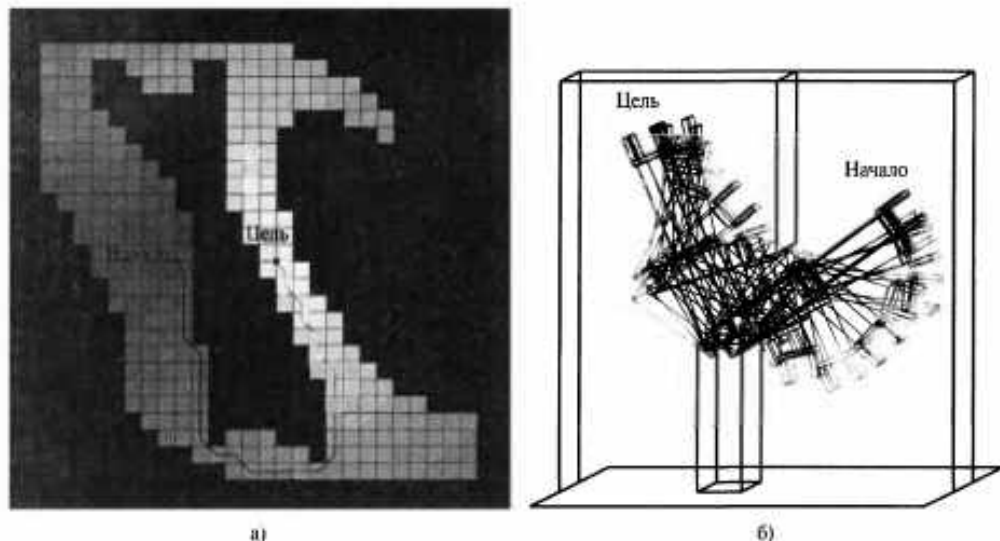


Рис. 25.13. Пример применения метода декомпозиции ячеек: функция стоимости и путь, найденный с помощью аппроксимации пространства конфигураций в виде ячеек сетки (а); тот же путь, визуализированный с помощью координат рабочего пространства (б). Обратите внимание на то, как робот сгибает свой локоть для предотвращения столкновения с вертикальным препятствием

Существуют два способа усовершенствования метода декомпозиции ячеек, позволяющие исправить эти недостатки. Первый из них состоит в том, что допускается дальнейшее разделение смешанных ячеек, возможно, с использованием ячеек, вдвое меньших по сравнению с первоначальным размером. Такая операция может продолжаться рекурсивно до тех пор, пока не будет найден путь, полностью проходящий по свободным ячейкам. (Безусловно, этот метод может применяться, только если есть возможность определить, является ли данная конкретная ячейка смешанной, а эта операция является простой, только если границы пространства конфигураций определяются с помощью относительно простых математических описаний.) Такой метод является полным, при условии, что заданы ограничения на величину наименьшего прохода, через который должен пройти искомый путь. Хотя при этом основная часть усилий, связанных с вычислениями, сосредоточивается на наиболее сложных участках в пространстве конфигураций, данный метод все еще не позволяет добиться успешного масштабирования и распространения его на многомерные задачи, поскольку при каждом рекурсивном разбиении ячейки создаются 2^d меньших ячеек. Второй способ получения полного алгоритма состоит в том, чтобы неуклонно соблюдалось требование **точной декомпозиции ячеек** свободного пространства. Этот метод должен допускать, чтобы ячейки принимали неправильную форму в тех местах, где они встречаются с границами свободного пространства, но эти формы все еще должны оставаться “простыми” в том смысле, что при их ис-

пользовании можно было легко вычислить траекторию прохождения через любую свободную ячейку. Для реализации этого метода требуется использование некоторых весьма сложных геометрических понятий, поэтому данный метод не будет рассматриваться здесь более подробно.

Рассматривая путь решения, показанный на рис. 25.13, а, можно заметить дополнительные сложности, которые необходимо преодолеть. Во-первых, следует отметить, что этот путь содержит произвольно острые углы; робот, движущийся с какой-либо конечной скоростью, не сможет пройти по такому пути. Во-вторых, заслуживает внимания то, что путь проходит очень близко от препятствия. Любой, кто занимается вождением автомобиля, знает, что парковочная площадка, на которой оставлено по одному миллиметру просвета с каждой стороны, в действительности вообще не годится для парковки; по той же причине следует предпочесть такие пути решения, которые не чувствительны к небольшим погрешностям движения.

Желательно максимизировать расстояние от препятствий и вместе с тем минимизировать длину пути. Этой цели можно достичь, введя понятие ∞ **поля потенциалов**. Поле потенциалов — это функция, определенная в пространстве состояний, значение которой растет пропорционально расстоянию до ближайшего препятствия. Такое поле потенциалов показано на рис. 25.14, а, — чем более темным цветом обозначена точка в пространстве конфигураций, тем ближе она к препятствию. При использовании в задаче планирования пути такое поле потенциалов становится дополнительным термом стоимости в уравнении оптимизации. Благодаря этому возникает интересная ситуация поиска компромисса. С одной стороны, робот стремится минимизировать длину пути к цели. С другой стороны, он пытается оставаться в стороне от препятствий, придерживаясь минимальных значений функции потенциалов. Назначив обоим целям соответствующие веса, можно найти примерно такой путь, как показано на рис. 25.14, б. На этом рисунке показана также функция стоимости, выведенная на основании комбинированной функции затрат, которая и в этом случае вычислена с помощью итерации по стоимостям. Очевидно, что полученный в результате путь длиннее, но вместе с тем и безопаснее.

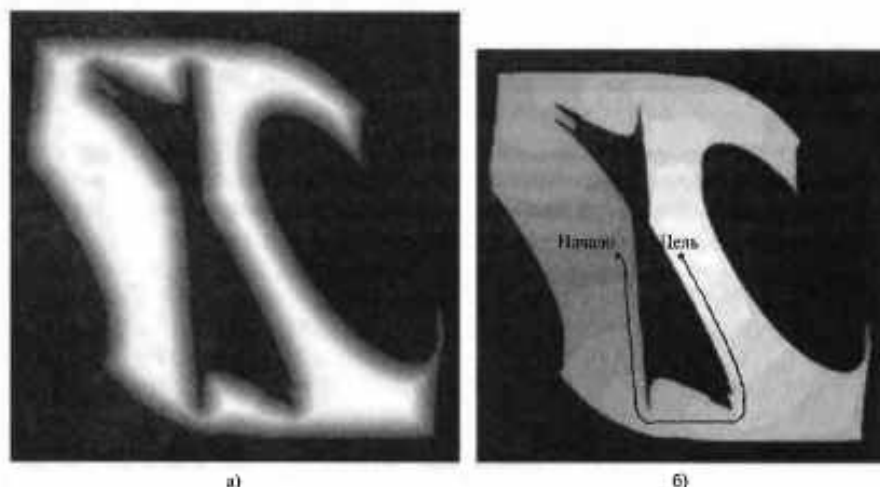


Рис. 25.14. Метод с использованием поля потенциалов: препятствующее сближению поле потенциалов отталкивает робота от препятствий (а); путь, найденный с помощью одновременной минимизации длины пути и потенциала (б)

Методы скелетирования

Второе важное семейство алгоритмов планирования пути основано на идее **скелетирования**. Эти алгоритмы сводят свободное пространство робота к одномерному представлению, для которого задача планирования становится проще. Такое представление с меньшим количеством измерений называется **скелетом** пространства конфигураций.

Пример применения метода скелетирования приведен на рис. 25.15: это — **линия Вороного** для свободного пространства, которая представляет собой геометрическое место всех точек, равноудаленных от двух или нескольких препятствий. Для того чтобы осуществить планирование пути с помощью линии Вороного, робот вначале переходит из текущей конфигурации в точку на линии Вороного. Можно легко показать, что такую операцию всегда можно выполнить с помощью передвижения по прямой в пространстве конфигураций. Затем робот следует по линии Вороного до тех пор, пока не достигнет точки, ближайшей к целевой конфигурации. Наконец, робот покидает линию Вороного и движется к цели. И на этом последнем этапе снова выполняется движение по прямой в пространстве конфигураций.

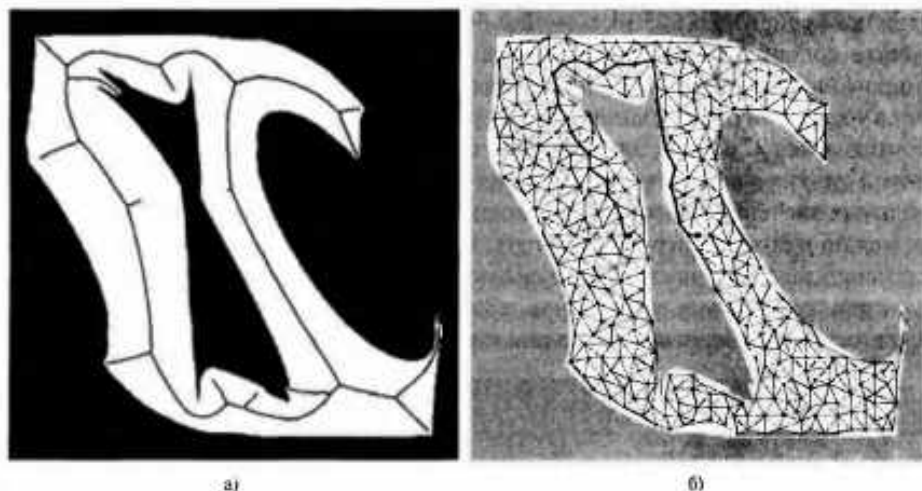


Рис. 25.15. Один из примеров метода скелетирования: линия Вороного — это геометрическое место точек, равноудаленных от двух или нескольких препятствий в пространстве конфигураций (а); вероятностная дорожная карта, состоящая из 400 случайно выбранных точек в свободном пространстве (б)

Таким образом, первоначальная задача планирования пути сводится к поиску пути на линии Вороного, которая обычно является одномерной (за исключением некоторых частных случаев) и имеет конечное количество таких точек, в которых пересекаются три или большее количество одномерных кривых. Поэтому задача поиска кратчайшего пути вдоль линии Вороного сводится к задаче поиска в дискретном графе такого же типа, как было описано в главах 3 и 4. Движение по линии Вороного может не обеспечить получение кратчайшего пути, но обнаруженные пути будут отличаться наличием максимальных расстояний от препятствий. Недостатки методов, основанных на использовании линии Вороного, состоят в том, что их сложно применять в пространствах конфигураций с большими размерностями, кроме того, при

их использовании приходится совершать слишком большие обходные маневры, если пространство конфигураций характеризуется широким размахом. К тому же может оказаться сложным вычисление линии Вороного, особенно в пространстве конфигураций, характеризующемся сложной формой препятствий.

Альтернативным по отношению к методу на основе линии Вороного является метод с использованием **вероятностной дорожной карты**. Он представляет собой такой подход к скелетированию, который позволяет определить больше возможных маршрутов и поэтому лучше подходит для пространств с широким размахом. Пример вероятностной дорожной карты показан на рис. 25.15, б. Линия, приведенная на этом рисунке, создана путем формирования случайным образом большого количества конфигураций и удаления тех из них, которые не укладываются в свободное пространство. После этого любые два узла соединяются какой-то линией, если одного из них можно “легко” достичь из другого; например, если в свободном пространстве можно перейти из одного узла в другой по прямой. Конечным итогом выполнения всех этих операций становится создание рандомизированного графа в свободном пространстве робота. Если к этому графу будут добавлены позиции начальной и целевой конфигураций робота, то задача планирования пути сведется к поиску в дискретном графе. Теоретически этот подход является неполным, поскольку при неудачном выборе случайно заданных точек может оказаться, что нельзя найти ни одного пути от начального узла до целевого. Но вероятность такой неудачи можно ограничить за счет регламентации количества формируемых точек и с учетом определенных геометрических свойств пространства конфигураций. Возможно также направить процесс выработки опорных точек в те области, где частично выполненный поиск показывает хорошие перспективы поиска приемлемого пути, действуя одновременно в двух направлениях, от начальной и от целевой позиций. После внесения всех этих усовершенствований метод планирования с помощью вероятностной дорожной карты показывает лучшую масштабируемость в условиях многомерных пространств конфигураций по сравнению с большинством других альтернативных методов планирования путей.

25.5. ПЛАНИРОВАНИЕ ДВИЖЕНИЙ В УСЛОВИЯХ НЕОПРЕДЕЛЕННОСТИ

Ни в одном из алгоритмов планирования движения робота, рассмотренных выше, не шла речь о наиболее важной характерной особенности робототехнических задач — об их неопределенности. В робототехнике неопределенность возникает из-за частичной наблюдаемости среды, а также под влиянием стохастических (или не предусмотренных моделью) результатов действий робота. Кроме того, могут возникать погрешности, обусловленные использованием приближенных алгоритмов, таких как фильтрация частиц, в результате чего робот не будет получать точных данных о текущем доверительном состоянии, даже несмотря на то, что для описания стохастического характера среды применяется идеальная модель.

В большинстве современных роботов для принятия решений используются детерминированные алгоритмы, такие как различные алгоритмы планирования пути, рассматривавшиеся до сих пор. Для этой цели обычно принято извлекать данные

о **наиболее вероятном состоянии** из распределения состояний, сформированного с помощью алгоритма локализации. Преимущество этого подхода состоит лишь в том, что он способствует уменьшению объема вычислений. Трудной является даже сама задача планирования путей через пространство конфигураций, а если бы нам пришлось работать с полным распределением вероятностей по состояниям, то задача стала бы еще труднее. Поэтому игнорировать неопределенность в этих обстоятельствах можно, только если неопределенность мала.

К сожалению, игнорировать неопределенность не всегда возможно. Дело в том, что при решении некоторых задач возникает такая ситуация, что неопределенность, в условиях которой действует робот, становится слишком большой. Например, как можно использовать детерминированный планировщик пути для управления мобильным роботом, не имеющим информации о том, где он находится? Вообще говоря, если истинное состояние робота не является таковым, на которое указывает правило максимального правдоподобия, то в итоге управляющие воздействия будут далеки от оптимальных. В зависимости от величины погрешности они могут приводить ко всякого рода нежелательным эффектам, таким как столкновения с препятствиями.

В этой области робототехники нашел свое применение целый ряд методов организации работы в условиях неопределенности. Некоторые из этих методов основаны на приведенных в главе 17 алгоритмах принятия решений в условиях неопределенности. Если робот сталкивается с неопределенностью только при переходах из одного состояния в другое, но само состояние является полностью наблюдаемым, то эту задачу лучше всего можно промоделировать в виде марковского процесса принятия решения, или MDP (Markov Decision Process). Решением задачи MDP является оптимальная **политика**, с помощью которой робот может определить, что делать в каждом возможном состоянии. Таким образом, он получает возможность исправить погрешности движения всех видов, тогда как решение, полученное от детерминированного планировщика, с указанием единственного пути, может быть гораздо менее надежным. В робототехнике вместо термина политика обычно используют термин **функция навигации**. Функцию стоимости, показанную на рис. 25.13, *а* можно преобразовать в такую функцию навигации, обеспечив отслеживание градиента.

Так же как и в задачах, описанных в главе 17, задачи, рассматриваемые в настоящей главе, становятся гораздо более трудными в условиях частичной наблюдательности. Возникающая в результате задача управления роботом представляет собой **частично наблюдаемую задачу MDP**, или POMDP (partially observable MDP). В таких ситуациях робот обычно поддерживает внутреннее доверительное состояние, наподобие описанного в разделе 25.3. Решением задачи POMDP является политика, определенная на доверительных состояниях робота. Иными словами, входными данными для рассматриваемой политики является все распределение вероятностей. Это позволяет роботу основывать свое решение не только на том, что ему известно, но и на том, что неизвестно. Например, если робот действует в условиях неопределенности в отношении какой-то важной переменной состояния, он может принять рациональное в этих условиях решение и вызвать на выполнение **действие по сбору информации**. Такой подход в инфраструктуре MDP невозможен, поскольку в задачах MDP подразумевается наличие полной наблюдаемости. К сожалению, методы точного решения задач POMDP не применимы к робототехнике, поскольку не существует известных методов для непрерывных пространств. А в результате дискретиза-

ции обычно создаются такие задачи POMDP, которые слишком велики, чтобы их можно было решить с помощью известных методов. Все, что можно сделать в настоящее время, — это пытаться свести неопределенность в отношении позы к минимуму; например, в эвристике **плавания вдоль берегов** требуется, чтобы робот оставался неподалеку от известных отметок в целях уменьшения неопределенности в отношении его позы. Такая ситуация, в свою очередь, приводит к постепенному уменьшению неопределенности при нанесении на карту обнаруженных поблизости новых отметок, а это в дальнейшем позволяет роботу исследовать новые территории.

Надежные методы

С неопределенностью можно также справиться, используя так называемые **надежные**, а не вероятностные методы. *Надежным* называется такой метод, в котором подразумевается наличие ограниченного объема неопределенности в каждом аспекте задачи, но не присваиваются вероятности значениям в пределах разрешенного интервала. *Надежным* называется такое решение, которое приводит к намеченной цели независимо от того, какие значения данных встречаются в действительности, при условии, что они находятся в пределах предполагаемого интервала. Крайней формой надежного метода является подход на основе **совместимого планирования**, описанный в главе 12, — в нем вырабатываются планы, выполнимые даже без учета информации о состоянии.

В настоящем разделе рассматривается один из надежных методов, применяемый для **планирования тонких движений** (или сокращенно FMP — Fine-Motion Planning) в задачах робототехнической сборки. Планирование тонких движений обеспечивает перемещение манипулятора робота в очень тесной близости от объекта в статической среде. Основная сложность, связанная с планированием тонких движений, состоит в том, что требуемые движения и соответствующие характеристики среды очень малы. В таких малых масштабах робот теряет возможность точно измерять или управлять своим положением, кроме того, может возникать неопределенность в отношении формы самой среды; предполагается, что все эти неопределенности ограничены. Решением задачи FMP обычно становится условный план (или политика), в котором используется обратная связь от датчиков и который гарантирует успешное выполнение во всех ситуациях, совместимых с предполагаемыми пределами неопределенности.

План проведения тонких движений представляет собой определение ряда **охраняемых движений**. Каждое охраняемое движение состоит, во-первых, из команды движения и, во-вторых, из условия завершения, которое представляет собой предикат, заданный на сенсорных значениях робота, и возвращает истинное значение в качестве указания на окончание охраняемого движения. Команды движения обычно задают **приспособляемые движения**, которые позволяют роботу выполнять скользящие движения, если другие команды движения вызовут столкновение с препятствием. В качестве примера на рис. 25.16 показано двухмерное пространство конфигураций с узким вертикальным отверстием. Такое пространство конфигураций может возникнуть при решении задачи вставки прямоугольного колышка в отверстие, немного превышающее его по размерам. Команды движения выполняются с постоянными скоростями. Условиями завершения являются ситуации контакта с поверхностью. Для моделирования неопределенности в процессе управления предположим, что факти-

чески движение робота происходит не в направлении, указанном в команде, а укладывается в конус C_v вокруг этого направления. На рис. 25.16 показано, что произойдет, если будет выдана команда движения с постоянной скоростью строго в вертикальном направлении из исходной области s . Из-за неопределенности в скорости робот может совершать движения в любом направлении в пределах конической огибающей; возможно, что это приведет к попаданию в отверстие, но с большей вероятностью колышек опустится с той или другой стороны от него. А поскольку робот не будет иметь информации о том, с какой стороны от отверстия опустился колышек, то не будет знать, куда его двигать дальше.

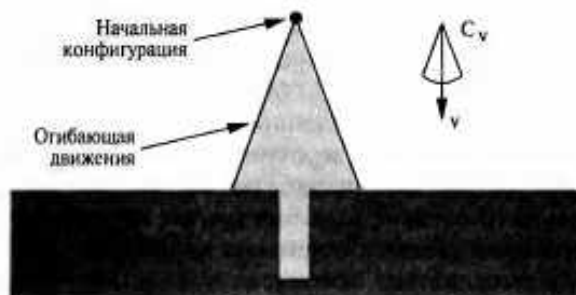


Рис. 25.16. Двухмерная среда, конус неопределенности скорости и огибающая возможных движений робота. Намеченная скорость равна v , но из-за неопределенности фактическая скорость может находиться в пределах C_v , а это приводит к тому, что окончательная конфигурация может определиться в любой точке внутри огибающей движения. Это означает, что возникает неопределенность в отношении того, удастся ли попасть в отверстие или нет

Более приемлемая стратегия показана на рис. 25.17 и 25.18. На рис. 25.17 приведен пример того, как робот намеренно направляет свое движение в определенную сторону от отверстия. Условия выполнения команды движения показаны на рисунке, а проверкой окончания движения становится контакт с любой поверхностью. На рис. 25.18 показано, что выдается команда движения, которая вынуждает робота передвигать колышек, скользящий по поверхности, до его попадания в отверстие. Тем самым предполагается использование команды приспособляемого движения. Поскольку все возможные скорости в огибающей движения направлены влево, робот должен передвинуть скользящий по поверхности колышек вправо после вступления его в контакт с горизонтальной поверхностью. Вслед за прикосновением колышка с правой вертикальной стенкой отверстия колышек должен проскользнуть вниз, поскольку все возможные скорости направлены вниз относительно этой вертикальной поверхности. Колышек будет продолжать двигаться до тех пор, пока не достигнет дна отверстия, поскольку именно таково условие завершения этого движения. Несмотря на неопределенность управления, все возможные траектории движения робота оканчиваются контактом с дном отверстия, разумеется, при условии, что не обнаружатся какие-либо дефекты поверхности, из-за которых робот не сможет сдвинуть с места застрявший колышек.

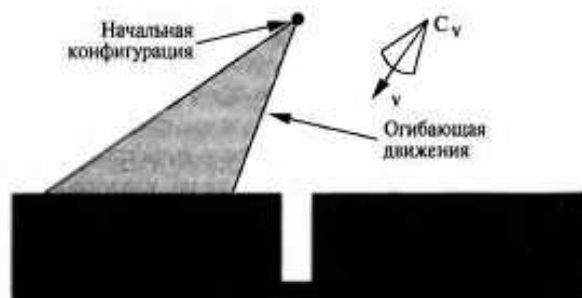


Рис. 25.17. Первая команда движения и полученная в итоге огибающая возможных движений робота. Независимо от любых погрешностей, известно, что в окончательной конфигурации колышек будет находиться слева от отверстия

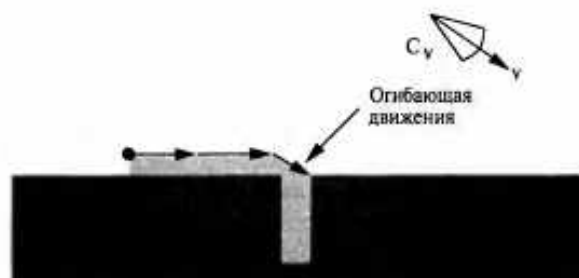


Рис. 25.18. Вторая команда движения и огибающая возможных движений. Даже при наличии погрешностей колышек в конечном итоге оказывается в отверстии

Вполне очевидно, что задача конструирования планов тонких движений не тривиальна; в действительности она намного сложнее по сравнению с задачей планирования в условиях точных движений. Для решения этой задачи можно либо выбирать постоянное количество дискретных значений для каждого движения, либо использовать геометрию среды для выбора направлений, позволяющих определить качественно иное поведение. Планировщик тонких движений принимает в качестве входных данных описание пространства конфигураций, угол наклона конуса неопределенности скоростей и спецификацию возможных сенсорных восприятий, определяющих ситуацию завершения (в данном случае — контакт с поверхностью). В свою очередь, планировщик должен выработать многоэтапный условный план (или политику), позволяющий добиться гарантированного успеха, если подобный план существует.

В рассматриваемом примере предполагается, что планировщик имеет в своем распоряжении точную модель среды, но возможно также принять допущение о наличии ограниченной погрешности в этой модели, как показано ниже. Если погрешность может быть описана в терминах параметров, то соответствующие параметры добавляются в качестве степеней свободы в пространство конфигураций. В частности, в последнем примере, если бы была неопределенность в отношении глубины и ширины отверстия, то можно было бы добавить эти величины в пространство конфигураций как две степени свободы. Из этого не следует, что появится возмож-

ность передвигать захват робота непосредственно в направлении этих степеней свободы в пространстве конфигураций или получать информацию об их позиции. Но оба эти ограничения можно учесть, описывая задачу как задачу FMP, задавая соответствующим образом данные о неопределенностях средств управления и датчиков. В результате возникает сложная, четырехмерная задача планирования, но появляется возможность применять точно такие же методы планирования, как и раньше. Следует отметить, что надежный подход такого рода приводит к созданию планов, в которых учитываются результаты самого неблагоприятного развития событий, а не максимизируется ожидаемое качество плана, в отличие от методов теории решений, описанных в главе 17. Единственным оптимальным аспектом планов действий в наиболее неблагоприятной ситуации (с точки зрения теории решений) является то, что они позволяют предотвратить последствия неудачи во время выполнения плана, намного худшие по сравнению с любыми другими затратами, связанными с его выполнением.

25.6. ОСУЩЕСТВЛЕНИЕ ДВИЖЕНИЙ

До сих пор речь в данной главе шла о том, как планировать движения, а не как их осуществлять. В разрабатываемых планах (особенно в тех, которые были составлены с помощью детерминированных планировщиков пути) предполагалось, что робот может просто проследовать по любому пути, сформированному алгоритмом. Но в реальном мире, безусловно, дело обстоит иначе. Роботы обладают инерцией и не могут выполнять произвольные команды движения по заданному пути, кроме как на произвольно низких скоростях. В большинстве случаев робот, выполняя команды движения, прилагает усилия для перемещения в ту или иную точку, а не просто задает нужные ему позиции. В данном разделе описаны методы вычисления таких усилий.

Динамика и управление

В разделе 25.2 введено понятие **динамического состояния**, которое расширяет представление о кинематическом состоянии робота, позволяя моделировать скорости робота. Например, в описании динамического состояния, кроме данных об угле поворота шарнира робота, отражена скорость изменения этого угла. В модели перехода для любого представления динамического состояния учитываются влияния усилий на эту скорость изменения. Подобные модели обычно выражаются с помощью **дифференциальных уравнений**, которые связывают количество (например, кинематическое состояние) с изменением этого количества во времени (например, скоростью). В принципе, можно было бы выбрать способ планирования движений робота с использованием динамических моделей вместо кинематических моделей, которые рассматривались в предыдущих разделах. Такая методология приводит к достижению превосходных показателей производительности робота, если удастся составить нужные планы. Однако динамическое состояние намного сложнее по сравнению с кинематическим пространством, а из-за большого количества измерений задачи планирования движений становятся неразрешимыми для любых роботов, кроме самых простых. По этой причине применяемые на практике робототехнические системы часто основаны на использовании более простых кинематических планировщиков пути.

Общепринятым методом компенсации ограничений кинематических планов является использование для слежения за роботом отдельного механизма, \approx **контроллера**. Контроллерами называются устройства, вырабатывающие команды управления роботом в реальном времени с использованием обратной связи от среды для достижения цели управления. Если цель состоит в удержании робота на заранее запланированном пути, то такие контроллеры часто называют \approx **опорными контроллерами**, а путь называют \approx **опорным путем**. Контроллеры, оптимизирующие глобальную функцию затрат, называют \approx **оптимальными контроллерами**. По существу, оптимальная политика для задачи MDP является определением оптимального контроллера.

На первый взгляд задача управления, позволяющая удерживать робот на заранее заданном пути, кажется относительно простой. Но на практике даже в ходе решения этой внешне простой задачи могут встретиться некоторые ловушки. На рис. 25.19, а показано, какие нарушения могут при этом возникать. На данном рисунке демонстрируется путь робота, предпринимającego попытку следовать по кинематическому пути. После возникновения любого отклонения (обусловленного либо шумом, либо ограничениями на те усилия, которые может применять робот) робот прикладывает противодействующее усилие, величина которого пропорциональна этому отклонению. Интуитивные представления говорят о том, что такой подход якобы вполне оправдан, поскольку отклонения должны компенсироваться противодействующим усилием, чтобы робот не отклонялся от своей траектории. Однако, как показано на рис. 25.19, а, действия такого контроллера вызывают довольно интенсивную вибрацию робота. Эта вибрация является результатом естественной инерции манипулятора робота — робот, резко направленный в стороны опорной позиции, проскакивает эту позицию, что приводит к возникновению симметричной погрешности с противоположным знаком. Согласно кривой, приведенной на рис. 25.19, а, такое перерегулирование может продолжаться вдоль всей траектории, поэтому результирующее движение робота далеко от идеального. Очевидно, что нужно предусмотреть лучший способ управления.

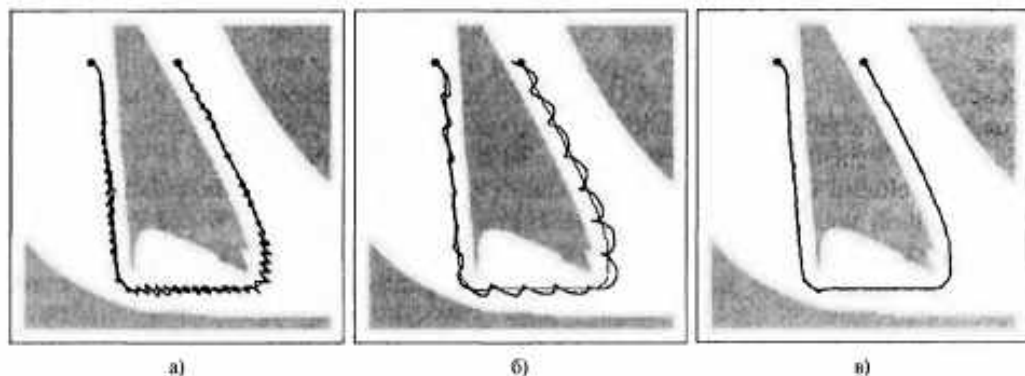


Рис. 25.19. Управление манипулятором робота с использованием различных методов: пропорциональное управление с коэффициентом усиления 1,0 (а); пропорциональное управление с коэффициентом усиления 0,1 (б); пропорционально-дифференциальное управление с коэффициентами усиления 0,3 для пропорционального и 0,8 для дифференциального компонента (в). Во всех случаях предпринимается попытка провести манипулятор робота по пути, обозначенному серым цветом

Для того чтобы понять, каким должен быть лучший контроллер, опишем формально тот тип контроллера, который допускает перерегулирование. Контроллеры, прикладывающие усилия, обратно пропорциональные наблюдаемой погрешности, называются **Р-контроллерами**. Буква Р является сокращением от proportional (пропорциональный) и показывает, что фактическое управляющее воздействие пропорционально погрешности позиционирования манипулятора робота. В качестве более формальной постановки допустим, что $y(t)$ — опорный путь, параметризованный временным индексом t . Управляющее воздействие a_t , выработанное Р-контроллером, имеет следующую форму:

$$a_t = K_P (y(t) - x_t)$$

где x_t — состояние робота во время t ; K_P — так называемый **коэффициент усиления** контроллера, от которого зависит, какое усилие будет прилагать контроллер, компенсируя отклонения между фактическим состоянием x_t и желаемым $y(t)$. В данном примере $K_P=1$. На первый взгляд может показаться, что проблему можно устранить, выбрав меньшее значение для K_P . Но, к сожалению, дело обстоит иначе. На рис. 25.19, б показана траектория манипулятора робота при $K_P=1$, в которой все еще проявляется колебательное поведение. Уменьшение величины коэффициента усиления способствует лишь уменьшению интенсивности колебаний, но не устраняет проблему. В действительности в отсутствие трения Р-контроллер действует в соответствии с законом пружины, поэтому он до бесконечности совершает колебания вокруг заданной целевой точки.

В традиционной науке задачи такого типа принадлежат к области **теории управления**, которая приобретает всю большую важность для исследователей в области искусственного интеллекта. Исследования в этой области, проводившиеся в течение десятков лет, привели к созданию многих типов контроллеров, намного превосходящих описанный выше контроллер, действующий на основании простого закона управления. В частности, опорный контроллер называется **стабильным**, если небольшие возмущения приводят к возникновению ограниченной погрешности, связывающей сигнал робота и опорный сигнал. Контроллер называется **строго стабильным**, если он способен вернуть управляемый им аппарат на опорный путь после воздействия подобных возмущений. Очевидно, что рассматриваемый здесь Р-контроллер только внешне кажется стабильным, но не является строго стабильным, поскольку не способен обеспечить возвращение робота на его опорную траекторию.

Простейший контроллер, позволяющий добиться строгой стабильности в условиях данной задачи, известен под названием **PD-контроллера**. Буква Р снова является сокращением от proportional (пропорциональный), а D — от derivative (дифференциальный). Для описания PD-контроллеров применяется следующее уравнение:

$$a_t = K_P (y(t) - x_t) + K_D \frac{\partial (y(t) - x_t)}{\partial t} \quad (25.3)$$

Согласно этому уравнению, PD-контроллеры представляют собой Р-контроллеры, дополненные дифференциальным компонентом, который добавляет к значению управляющего воздействия a_t терм, пропорциональный первой производной от погрешности $y(t) - x_t$ по времени. К какому результату приводит добавление такого терма? Вообще говоря, дифференциальный терм гасит колебания в системе, для управле-

ния которой он применяется. Чтобы убедиться в этом, рассмотрим ситуацию, в которой погрешность $(y(t) - x_t)$ резко изменяется во времени, как было в случае с Р-контроллером, описанным выше. При этом производная такой погрешности прикладывается в направлении, противоположном пропорциональному терму, что приводит к уменьшению общего отклика на возмущение. Однако, если та же погрешность продолжит свое присутствие и не изменится, то производная уменьшится до нуля и при выборе управляющего воздействия будет доминировать пропорциональный терм.

Результаты применения такого PD-контроллера для управления манипулятором робота при использовании в качестве коэффициентов усиления значений $K_p = .3$ и $K_D = .8$ показаны на рис. 25.19, в. Очевидно, что в конечном итоге траектория манипулятора стала гораздо более гладкой и на ней внешне не заметны какие-либо колебания. Как показывает этот пример, дифференциальный терм позволяет обеспечить стабильность работы контроллера в тех условиях, когда она недостижима другими способами.

Но, как показывает практика, PD-контроллеры также создают предпосылки отказов. В частности, PD-контроллеры могут оказаться неспособными отрегулировать погрешность до нуля, даже при отсутствии внешних возмущений. Такой вывод не следует из приведенного в этом разделе примера робота, но, как оказалось, иногда для уменьшения погрешности до нуля требуется приложить обратную связь с пропорциональным перерегулированием. Решение этой проблемы заключается в том, что к закону управления нужно добавить третий терм, основанный на результатах интегрирования погрешности по времени:

$$a_t = K_p (y(t) - x_t) + K_I \int (y(t) - x_t) dt + K_D \frac{\partial (y(t) - x_t)}{\partial t} \quad (25.4)$$

где K_I — еще один коэффициент усиления. В терме

$$\int (y(t) - x_t) dt$$

вычисляется интеграл погрешности по времени. Под влиянием этого терма корректируется продолжающиеся долгое время отклонения между опорным сигналом и фактическим состоянием. Если, например, x_t меньше чем $y(t)$ в течение продолжительного периода времени, то значение этого интеграла возрастает до тех пор, пока результирующее управляющее воздействие a_t не вызовет уменьшение этой погрешности. Таким образом, интегральные термы гарантируют отсутствие систематических погрешностей в действиях контроллера за счет повышения опасности колебательного поведения. Контроллер, закон управления которого состоит из всех трех термов, называется **ПИД-контроллером**. ПИД-контроллеры широко используются в промышленности для решения самых различных задач управления.

Управление на основе поля потенциалов

Выше в данной главе поле потенциалов было определено как дополнительная функция затрат в планировании движений робота, но поле потенциалов может также использоваться для непосредственной выработки траектории движения робота,

что позволяет полностью отказаться от этапа планирования пути. Для достижения этой цели необходимо определить притягивающее усилие, которое влечет манипулятор робота в направлении его целевой конфигурации, и поле потенциалов, отталкивающее манипулятор, которое отводит манипулятор от препятствий. Такое поле потенциалов показано на рис. 25.20. Его единственным глобальным минимумом является целевая конфигурация, а стоимость измеряется суммой расстояния до целевой конфигурации и дальности до препятствий. Для формирования поля потенциалов, показанного на этом рисунке, не требовалось никакого планирования. Благодаря такой их особенности поля потенциалов могут успешно применяться в управлении в реальном времени. На рис. 25.20 показаны две траектории робота, осуществляющего восхождение к вершине в поле потенциалов при двух различных начальных конфигурациях. Во многих приложениях поле потенциалов может быть эффективно рассчитано для любой конкретной конфигурации. Кроме того, оптимизация потенциала сводится к вычислению градиента потенциала для текущей конфигурации робота. Такие вычисления обычно являются чрезвычайно эффективными, особенно по сравнению с алгоритмами планирования пути, которые связаны с затратами, характеризующимися экспоненциальной зависимостью от размерностей пространства конфигураций (от степеней свободы).

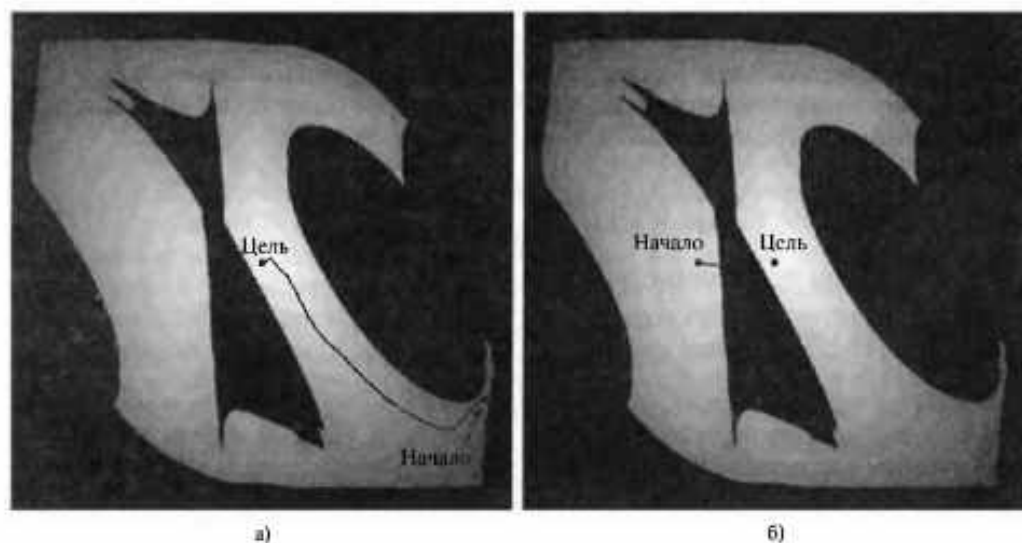




Рис. 25.20. Метод управления на основе поля потенциалов. Траектория робота восходит по градиенту поля потенциалов, состоящего из отталкивающих усилий, обусловленных наличием препятствий, и притягивающих усилий, которые соответствуют целевой конфигурации: успешно пройденный путь (а); локальный оптимум (б)

Тот факт, что подход на основе поля потенциалов позволяет так эффективно находить путь к цели, даже если при этом приходится преодолевать большие расстояния в пространстве конфигураций, ставит под вопрос саму целесообразность использования планирования в робототехнике. Действительно ли методы на основе поля потенциалов позволяют решать все задачи, или такая ситуация, как в данном примере, является просто результатом благоприятного стечения обстоятельств? Ответ заключается в том, что в данном случае обстоятельства действительно складыва-

лись благоприятно, а в других условиях поля потенциалов имеют много локальных минимумов, которые могут стать ловушкой для робота. В данном примере робот приближается к препятствию, вращая шарнир своего плеча до тех пор, пока не заходит в тупик, оказавшись не с той стороны препятствия, с какой нужно. Поле потенциалов не имеет достаточно развитой конфигурации для того, чтобы вынудить робот согнуть свой локоть, что позволило бы ему пропустить манипулятор под препятствием. Иными словами, методы на основе поля потенциалов превосходно подходят для локального управления роботом, но все еще требуют глобального планирования. Еще одним важным недостатком, связанным с использованием поля потенциалов, является то, что усилия, вырабатываемые при этом подходе, зависят только от положений препятствия и робота, а не от скорости робота. Таким образом, метод управления на основе поля потенциалов в действительности является кинематическим и может оказаться неприменимым для быстро движущегося робота.

Реактивное управление

До сих пор в этой главе речь шла об управляющих решениях, которые требуют наличия определенной модели среды, чтобы на ее основе можно было сформировать либо опорный путь, либо поле потенциалов. Но с этим подходом связаны некоторые сложности. Во-первых, зачастую сложно получить достаточно точные модели, особенно в сложной или удаленной среде, такой как поверхность Марса. Во-вторых, даже в тех случаях, когда есть возможность составить модель с достаточной точностью, вычислительные сложности и погрешности локализации могут привести к тому, что эти методы окажутся практически не применимыми. В определенных обстоятельствах более подходящим становится один из видов рефлексного проекта агента — проект на основе так называемого  **реактивного управления**.

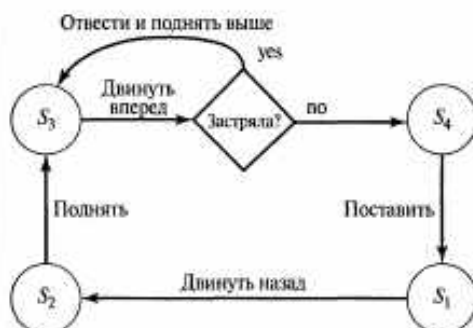
Одним из примеров такого проекта является шестиногий робот, или  **гексапод**, показанный на рис. 25.21, а, который предназначен для ходьбы по пересеченной местности. В целом датчики робота не позволяют формировать модели местности с точностью, достаточной для любого из методов планирования пути, описанных в предыдущем разделе. Кроме того, даже в случае использования достаточно точных датчиков задача планирования пути не разрешима с помощью имеющихся вычислительных средств из-за наличия двенадцати степеней свободы (по две для каждой ноги).

Тем не менее существует возможность определить спецификацию контроллера непосредственно, без использования явной модели среды. (Выше в данной главе такой подход уже был продемонстрирован на примере PD-контроллера, который оказался способным вести сложный манипулятор робота к цели при отсутствии явной модели динамики робота; однако для этого контроллера требовался опорный путь, сформированный с помощью кинематической модели.) Для рассматриваемого примера шагающего робота после выбора подходящего уровня абстракции задача определения закона управления оказалась удивительно простой. В приемлемом законе управления может быть предусмотрено циклическое движение каждой ноги с тем, чтобы эта нога на какой-то момент касалась земли, а в остальное время двигалась в воздухе. Координация действий всех шести ног должна осуществляться так, чтобы три из них (расположенные на противоположных концах) всегда находились на земле для обеспечения физической опоры. Такой принцип управления можно легко запрограммировать, и он себя полностью оправдывает на ровной местности. А на пе-

ресеченной местности движению ног вперед могут помешать препятствия. Это затруднение можно преодолеть с помощью исключительно простого правила управления: если движение какой-то ноги вперед блокируется, следует отвести ее немного назад, поднять выше и предпринять еще одну попытку. Созданный в итоге контроллер показан на рис. 25.21, б в виде конечного автомата; он представляет собой рефлексный агент с поддержкой состояния, в котором внутреннее состояние представлено индексом текущего состояния автомата (от s_1 до s_4).



а)



б)

Рис. 25.21. Пример применения реактивного управления: шестиногий робот (а); дополненный конечный автомат (Augmented Finite State Machine — AFSM) для управления одной ногой (б). Обратите внимание на то, что этот автомат AFSM реагирует на сенсорную обратную связь: если какая-то нога не может двинуться вперед при выполнении этапа ее поворота и переноса в прямом направлении, то она поднимается каждый раз все выше и выше

Практика показала, что разновидности такого простого контроллера, действующего на основе обратной связи, позволяют реализовывать исключительно надежные способы ходьбы, с помощью которых робот свободно маневрирует на пересеченной местности. Очевидно, что в таком контроллере не используется модель, кроме того, для выработки управляющих воздействий не осуществляется алгоритмический вывод и не производится поиск. В процессе эксплуатации подобного контроллера решающую роль в выработке поведения роботом играет обратная связь от среды. Само программное обеспечение робота, отдельно взятое, не определяет, что фактически происходит после того, как робот входит в какую-то среду. Поведение, проявляющееся в результате взаимодействия (простого) контроллера и (сложной) среды, часто называют **эмерджентным поведением** (т.е. поведением не планируемым, а обусловленным ситуацией). Строго говоря, все роботы, рассматриваемые в этой главе, обнаруживают эмерджентное поведение в связи с тем фактом, что ни одна из используемых в них моделей не является идеальной. Но по традиции этот термин применяется для обозначения лишь таких методов управления, в которых не используются явно заданные модели среды. Кроме того, эмерджентное поведение является характерным для значительной части биологических организмов.

С формальной точки зрения реактивные контроллеры представляют собой одну из форм реализации политики для задач MDP (или, если они имеют внутреннее состояние, для задач POMDP). В главе 17 было описано несколько методов выработки политики на основании модели робота и его среды. В робототехнике большое практическое значение имеет подход, предусматривающий составление подобной политики

вручную, поскольку часто невозможно сформулировать точную модель. В главе 21 описаны методы обучения с подкреплением, позволяющие формировать политику на основании опыта. Некоторые из подобных методов (такие как Q-обучение и поиск политики) не требуют модели среды и позволяют создавать высококачественные контроллеры для роботов, но взамен требуют предоставления огромных объемов обучающих данных.

25.7. АРХИТЕКТУРЫ РОБОТОТЕХНИЧЕСКОГО ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ

✎ **Архитектурой программного обеспечения** называется применяемая методология структуризации алгоритмов. В понятие архитектуры обычно включают языки и инструментальные средства для написания программ, а также общий подход к тому, как должно быть организовано взаимодействие программ.

Предложения по созданию современных архитектур программного обеспечения для робототехники должны отвечать на вопрос о том, как сочетаются реактивные средства управления и основанные на модели алгоритмические средства управления. Преимущества и недостатки реактивных и алгоритмических средств управления таковы, что эти средства во многом дополняют друг друга. Реактивные средства управления действуют на основании сигналов, полученных от датчиков, и хорошо подходят для принятия решений низкого уровня в реальном времени. Однако реактивные средства управления лишь в редких случаях позволяют вырабатывать приемлемые решения на глобальном уровне, поскольку глобальные управляющие решения зависят от информации, которая не может быть воспринята с помощью датчиков во время принятия решения. Для таких задач в большей степени подходят алгоритмические средства управления.

В связи с этим в большинстве вариантов робототехнических архитектур реактивные методы используются на низких уровнях управления, а алгоритмические методы — на более высоких уровнях. Такая комбинация средств управления уже встречалась в данной главе в описании PD-контроллеров, где было показано, как сочетается (реактивный) PD-контроллер с (алгоритмическим) планировщиком пути. Архитектуры, в которых сочетаются реактивные и алгоритмические методы, обычно называют ✎ **гибридными архитектурами**.

Обобщающая архитектура

✎ **Обобщающая архитектура** [189] представляет собой инфраструктуру для сборки реактивных контроллеров из конечных автоматов. Узлы этих автоматов могут содержать средства проверки для некоторых сенсорных переменных, и в таких случаях трассировка выполнения конечного автомата становится обусловленной результатом подобной проверки. Дуги могут быть размечены сообщениями, которые вырабатываются при прохождении по этим дугам, после чего сообщения передаются в двигатели робота или другие конечные автоматы. Кроме того, конечные автоматы оборудованы внутренними таймерами (часами), которые управляют затратами времени, требуемыми для прохождения дуги. Полученные в итоге автоматы обычно на-

зывают **дополненными конечными автоматами**, или сокращенно AFSM (Augmented Finite State Machine), где под дополнением подразумевается использование часов.

Примером простого автомата AFSM может служить автомат с четырьмя состояниями, показанный на рис. 25.21, б, который вырабатывает команды циклического движения ноги для шестиногого шагающего робота. Этот автомат AFSM реализует циклический контроллер, который действует в основном без учета обратной связи от среды. Тем не менее обратная связь от датчика учитывается на этапе переноса ноги вперед. Если нога не может пройти вперед, что означает наличие впереди препятствия, робот отводит ногу немного назад, поднимает ее выше и предпринимает попытку еще раз выполнить шаг вперед. Поэтому данный контроллер приобретает способность реагировать на непредвиденные ситуации, возникающие в процессе взаимодействия робота со своей средой.

В обобщающей архитектуре предусмотрены дополнительные примитивы для синхронизации работы автоматов AFSM, а также для комбинирования выходных данных, поступающих от многочисленных, возможно конфликтующих автоматов AFSM. Благодаря этому такая архитектура позволяет программисту компоновать все более и более сложные контроллеры по принципу восходящего проектирования. В рассматриваемом примере можно начать с создания автоматов AFSM для отдельных ног, а вслед за этим ввести автомат AFSM для координации действий одновременно нескольких ног. В качестве надстройки над этими автоматами можно реализовать такое поведение высокого уровня, как предотвращение столкновений с крупными препятствиями, для чего могут потребоваться такие операции, как отступление и изменение направления движения.

Идея компоновки контроллеров роботов из автоматов AFSM весьма заманчива. Достаточно представить себе, насколько сложно было бы выработать такое же поведение с использованием любого из алгоритмов планирования пути в пространстве конфигураций, описанных в предыдущем разделе. Прежде всего, потребовалось бы точная модель местности. Пространство конфигураций робота с шестью ногами, каждая из которых приводится в действие двумя отдельными двигателями, имеет общее количество измерений, равное восемнадцати (двенадцать измерений для конфигурации ног и шесть для локализации и ориентации робота относительно его среды). Даже если бы применяемые компьютеры были достаточно быстродействующими для того, чтобы обеспечить поиск путей в таких многомерных пространствах, все равно нельзя было бы избавиться от необходимости учитывать такие неприятные ситуации, как скольжение робота по склону. Из-за таких стохастических эффектов единственный найденный путь через пространство конфигураций был бы наверняка слишком трудновыполнимым, и с возможными непредвиденными ситуациями не помог бы справиться даже PID-контроллер. Иными словами, задача выработки поведения, определяющего движение с помощью алгоритмических расчетов, слишком сложна для современных алгоритмов планирования движения робота.

К сожалению, обобщающая архитектура также не лишена недостатков. Во-первых, автоматы AFSM обычно действуют под управлением непосредственно полученных сенсорных входных данных. Такая организация работы оправдывает себя, если сенсорные данные надежны и содержат всю необходимую информацию для принятия решения, но становится неприемлемой, если есть необходимость агрегировать сенсорные данные, полученные за определенные промежутки времени, с помощью нетривиальных способов. Поэтому контроллеры обобщающего типа в основном

применяются для решения локальных задач, таких как прохождение вдоль стены или движение к видимым источникам света. Во-вторых, из-за отсутствия средств выполнения алгоритмических расчетов изменение задания для робота становится затруднительным. Робот обобщающего типа обычно выполняет только одно задание, и нет такого способа, который позволил бы модифицировать его средства управления таким образом, чтобы он приспособился к выполнению других задач управления (в этом роботы такого типа полностью аналогичны навозному жуку, о котором речь шла на с. 1). Наконец, в работе контроллеров обобщающего типа трудно разобраться. На практике возникают такие запутанные взаимодействия десятков одновременно функционирующих автоматов AFSM (и со средой), что большинство программистов не в состоянии их проанализировать. По всем этим причинам, несмотря на свою огромную историческую важность, обобщающая архитектура редко используется в коммерческой робототехнике. Но для некоторых ее потомков судьба сложилась иначе.

Трехуровневая архитектура

В гибридных архитектурах реакции объединяются с алгоритмическими вычислениями. Одним из видов гибридной архитектуры, намного превосходящим другие по своей популярности, является **трехуровневая архитектура**, которая состоит из реактивного, исполнительного и алгоритмического уровней.

➤ **Реактивный уровень** обеспечивает низкоуровневое управление роботом. Отличительной особенностью этого уровня является наличие жесткого цикла “восприятие—действие”. Время принятия решения на этом уровне часто составляет лишь несколько миллисекунд.

➤ **Исполнительный уровень** (или уровень упорядочения) служит в качестве посредника между реактивным и алгоритмическим уровнями. Он принимает директивы от алгоритмического уровня и упорядочивает их для передачи на реактивный уровень. Например, на исполнительном уровне может быть выполнена обработка множества промежуточных пунктов, сформированного алгоритмическим планировщиком пути, а затем приняты решения о том, какое реактивное поведение должно быть вызвано. Время принятия решения на исполнительном уровне обычно составляет порядка одной секунды. Исполнительный уровень отвечает также за интеграцию сенсорной информации в виде представления внутреннего состояния. Например, на этом уровне могут быть реализованы процедуры локализации робота и оперативного составления карты.

На **алгоритмическом уровне** вырабатываются глобальные решения сложных задач с использованием методов планирования. Из-за вычислительной сложности, связанной с выработкой таких решений, время принятия решений на этом уровне составляет порядка нескольких минут. На алгоритмическом уровне (или уровне планирования) для принятия решений используются модели. Эти модели могут быть подготовлены заранее или сформированы путем обучения на основе имеющихся данных, и в них обычно используется информация о состоянии, собранная на исполнительном уровне.

В большинстве современных систем робототехнического программного обеспечения используются те или иные варианты трехуровневой архитектуры. Но критерии декомпозиции на три уровня нельзя назвать очень строгими. К тому же в неко-

торых системах робототехнического программного обеспечения предусмотрены дополнительные уровни, такие как уровни пользовательского интерфейса, которые управляют взаимодействием с пользователями, или уровни, ответственные за координацию действий робота с действиями других роботов, функционирующих в той же среде.

Робототехнические языки программирования

Многие робототехнические контроллеры реализованы с использованием языков программирования специального назначения. Например, многие программы для обобщающей архитектуры были реализованы на **языке поведения**, который был определен Бруксом [191]. Этот язык представляет собой язык управления в реальном времени на основе правил, результатом компиляции которого становятся контроллеры AFSM. Отдельные правила этого языка, заданные с помощью синтаксиса, подобного Lisp, компилируются в автоматы AFSM, а группы автоматов AFSM объединяются с помощью совокупности механизмов передачи локальных и глобальных сообщений.

Так же как и обобщающая архитектура, язык поведения является ограниченным, поскольку он нацелен на создание простых автоматов AFSM с относительно узким определением потока связи между модулями. Но в последнее время на базе этой идеи проведены новые исследования, которые привели к созданию целого ряда языков программирования, аналогичных по своему духу языку поведения, но более мощных и обеспечивающих более быстрое выполнение. Одним из таких языков является **универсальный робототехнический язык**, или сокращенно GRL (Generic Robot Language) [681]. GRL — это функциональный язык программирования для создания больших модульных систем управления. Как и в языке поведения, в GRL в качестве основных конструктивных блоков используются конечные автоматы. Но в качестве настройки над этими автоматами язык GRL предлагает гораздо более широкий перечень конструкций для определения коммуникационного потока и синхронизации ограничений между различными модулями, чем язык поведения. Программы на языке GRL компилируются в эффективные программы на таких языках команд, как C.

Еще одним важным языком программирования (и связанной с ним архитектурой) для параллельного робототехнического программного обеспечения является система планирования реактивных действий, или сокращенно RAPS (Reactive Action Plan System) [470]. Система RAPS позволяет программистам задавать цели, планы, связанные с этими целями (или частично определять политику), а также задавать условия, при которых эти планы по всей вероятности будут выполнены успешно. Крайне важно то, что в системе RAPS предусмотрены также средства, позволяющие справиться с неизбежными отказами, которые возникают в реальных робототехнических системах. Программист может задавать процедуры обнаружения отказов различных типов и предусматривать процедуру устранения исключительной ситуации для каждого типа отказа. В трехуровневых архитектурах система RAPS часто используется на исполнительном уровне, что позволяет успешно справляться с непредвиденными ситуациями, не требующими перепланирования.

Существует также несколько других языков, которые обеспечивают использование в роботах средств формирования рассуждений и средств обучения. Например, Golob [918] представляет собой язык программирования, позволяющий обеспечить безукоризненное взаимодействие средств алгоритмического решения задач (планирования) и средств реактивного управления, заданных непосредственно с помощью специфика-

ции. Программы на языке Golog формулируются в терминах ситуационного исчисления (см. раздел 10.3) с учетом дополнительной возможности применения операторов недетерминированных действий. Кроме спецификации программы управления с возможностями недетерминированных действий, программист должен также предоставить полную модель робота и его среды. Как только программа управления достигает точки недетерминированного выбора, вызывается планировщик (заданный в форме программы доказательства теорем) для определения того, что делать дальше. Таким образом программист может определять частично заданные контроллеры и опираться на использование встроенных планировщиков для принятия окончательного выбора плана управления. Основной привлекательной особенностью языка Golog является предусмотренная в нем безукоризненная интеграция средств реактивного управления и алгоритмического управления. Несмотря на то что при использовании языка Golog приходится соблюдать строгие требования (полная наблюдаемость, дискретные состояния, полная модель), с помощью этого языка были созданы высокоуровневые средства управления для целого ряда мобильных роботов, предназначенных для применения внутри помещений.

Язык \propto CES (сокращение от C++ for embedded systems — C++ для встроенных систем) — это языковое расширение C++, в котором объединяются вероятностные средства и средства обучения [1507]. В число типов данных CES входят распределения вероятностей, что позволяет программисту проводить расчеты с использованием неопределенной информации, не затрачивая тех усилий, которые обычно связаны с реализацией вероятностных методов. Еще более важно то, что язык CES обеспечивает настройку робототехнического программного обеспечения с помощью обучения на основании примеров, во многом аналогично тому, что осуществляется в алгоритмах обучения, описанных в главе 20. Язык CES позволяет программистам оставлять в коде “промежутки”, которые заполняются обучающими функциями; обычно такими промежутками являются дифференцируемые параметрические представления, такие как нейронные сети. В дальнейшем на отдельных этапах обучения, для которых учитель должен задать требуемое выходное поведение, происходит индуктивное обучение с помощью этих функций. Практика показала, что язык CES может успешно применяться в проблемных областях, характерных для частично наблюдаемой и непрерывной среды.

Язык \propto ALisp [32] представляет собой расширение языка Lisp. Язык ALisp позволяет программистам задавать недетерминированные точки выбора, аналогичные точкам выбора в языке Golog. Но в языке ALisp для принятия решений применяется не программа доказательства теорем, а средства определения правильного действия с помощью индуктивного обучения, в которых используется обучение с подкреплением. Поэтому язык ALisp может рассматриваться как удобный способ внедрения знаний о проблемной области в процедуру обучения с подкреплением, особенно знаний об иерархической структуре “процедур” желаемого поведения. До сих пор язык ALisp применялся для решения задач робототехники только в имитационных исследованиях, но может стать основой многообещающей методологии создания роботов, способных к обучению в результате взаимодействия со своей средой.

25.8. ПРИКЛАДНЫЕ ОБЛАСТИ

В настоящем разделе описаны некоторые из основных областей приложения для робототехнической технологии.

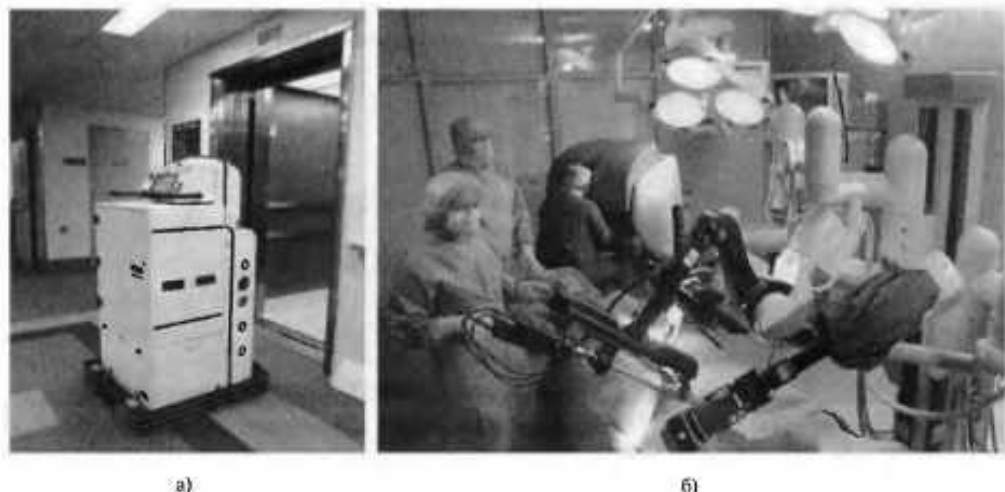
- **Промышленность и сельское хозяйство.** Роботы издавна предназначались для использования в тех областях, где требуется тяжелый труд, но трудовые процессы достаточно структурированы, чтобы допускать возможность робототехнической автоматизации. Самым удачным примером может служить сборочная линия, на которой манипуляторы уже давно выполняют такие задачи, как сборка, установка деталей, доставка материалов, сварка и окраска. При решении многих из этих задач роботы оказались более экономически эффективными по сравнению с работниками-людьми.

Для использования на открытых площадках конструктивное исполнение в виде роботов получили многие из тяжелых машин, которые применяются для сбора урожая, подземной проходки или рытья котлованов. Например, в университете Carnegie—Mellon недавно был завершен проект, который показал, что роботы могут использоваться для очистки от старой краски корпусов крупных судов, причем выполняют эту операцию в 50 раз быстрее, чем люди, и оказывают гораздо меньшее вредное воздействие на окружающую среду. Кроме того, было показано, что прототипы автономных роботов-проходчиков работают быстрее и точнее, чем люди, при выполнении задач по транспортировке руды в подземных шахтах. Роботы использовались для составления карт заброшенных шахт и канализационных систем с высокой точностью. Хотя проекты многих из этих систем все еще находятся на этапе разработки их прототипов, наступление той эпохи, когда роботы возьмут на себя основную часть полумеханической работы, которая в настоящее время выполняется людьми, — лишь вопрос времени.

- **Транспортировка.** Области применения робототехнических систем транспортировки являются весьма разнообразными, начиная от автономных вертолетов, которые доставляют объекты в те места, доступ к которым трудно получить иными способами, и заканчивая автоматическими инвалидными колясками, которые перевозят людей, не способных самостоятельно управлять этими колясками, или автономными порталными погрузчиками, которые превосходят самых опытных крановщиков по точности доставки контейнеров с судов на автомобили в грузовых доках. Одним из самых наглядных примеров успешно действующих транспортных роботов, предназначенных для работы внутри помещения, называемых также гоферами (gopher — мальчик на побегушках), является робот Helpmate, показанный на рис. 25.22, *а*. Такие роботы применяются в десятках больниц для транспортировки пищи и других медицинских материалов. Исследователи разработали робототехнические системы, напоминающие автомобили, которые способны автономно передвигаться по автомагистралям или по бездорожью. В условиях промышленных предприятий автономные транспортные средства в настоящее время стали одним из обычных средств доставки материалов на складах и производственных линиях.

Для эксплуатации многих из этих роботов требуется внести в среду их применения определенные модификации. К числу наиболее распространенных модификаций относятся средства локализации, такие как индуктивные контуры в полу, активные маяки, метки в виде штрих-кода и спутники системы GPS. Поэтому нерешенной задачей в робототехнике является проектирование роботов, способных использовать для своей навигации не искусственные при-

способления, а естественные признаки, особенно в таких условиях, как дальнее океанское плавание, где система GPS недоступна.



а)

б)

*Рис. 25.22. Примеры применения роботов в медицине: робот *Helpmate* применяется для транспортировки пищи и других медицинских препаратов в десятках больниц во всем мире (а); хирургические роботы в операционной (предоставлено компанией *da Vinci Surgical Systems*) (б)*

- **Работа в опасной среде.** Роботы помогали людям при очистке ядерных отходов. К числу наиболее ярких примеров относится устранение последствий аварии в Чернобыле и на острове Тримайл Айленд. Роботы применялись для расчистки завалов Всемирного торгового центра в Нью-Йорке и направлялись на те участки, которые считались слишком опасными для поисковых и спасательных команд.

В некоторых странах роботы используются для транспортировки взрывчатых веществ и обезвреживания бомб; как известно, эти работы характеризуются повышенной опасностью. В настоящее время во многих исследовательских проектах ведется разработка прототипов роботов для обезвреживания минных полей на земле и на море. Большинство существующих роботов для таких задач действуют в режиме телеуправления — ими управляет оператор с помощью средств дистанционного управления. Следующим важным шагом является предоставление таким роботам автономии.

- **Разведка.** Роботы добрались до тех мест, где еще никто не бывал, включая поверхность Марса (см. рис. 25.1, а). С помощью роботов-манипуляторов космонавты выводят на орбиту и снимают с орбиты спутники, а также строят Международную космическую станцию. Кроме того, роботы помогают проводить подводные исследования. Например, с использованием роботов уже было составлено много карт расположения затонувших кораблей. На рис. 25.23 показан робот, составляющий карту заброшенной угольной шахты, наряду с трехмерной моделью карты, формируемой с помощью датчиков расстояния. В 1996 году группа исследователей выпустила шагающего робота в кратер действующего вулкана, чтобы он собрал данные, важные для климатических ис-

следований. В военных операциях широко используются автономные воздушные транспортные средства, известные под названием **беспилотных самолетов**. Роботы становятся очень эффективными средствами сбора информации в тех областях, доступ к которым является сложным (или опасным) для людей.



а)



б)

Рис. 25.23. Примеры применения роботов в исследованиях труднодоступных участков: робот, используемый для составления карты заброшенной угольной шахты (а); трехмерная карта угольной шахты, составленная роботом (б)

- **Здравоохранение.** Роботы все чаще используются в качестве помощников хирургов при проведении операций на таких важных органах, как мозг, глаза и сердце. На рис. 25.22, б показана подобная система. Благодаря их высокой точности роботы стали незаменимым инструментальным средством при выполнении некоторых видов операций по эндопротезированию тазобедренного сустава. В экспериментальных исследованиях было обнаружено, что робототехнические устройства снижают опасность повреждений при выполнении колоноскопии. За пределами операционной исследователи стали разрабатывать средства робототехнической помощи для пожилых людей и инвалидов, такие как интеллектуальные автоматизированные шагающие аппараты и интеллектуальные игрушки, которые напоминают о наступлении времени приема лекарства.
- **Персональные услуги.** Перспективной областью применения робототехники становится предоставление услуг. Роботы-ассистенты помогают людям выполнять повседневные задачи. В число коммерчески доступных роботов-помощников по дому входят автономные пылесосы, газонокосилки и подносчики ключей для гольфа. Все эти роботы способны передвигаться самостоятельно и выполняют свои задачи без помощи людей. Некоторые обслуживающие роботы функционируют в общественных местах; к таким примерам относятся роботизированные справочные бюро, развернутые в крупных универсаме и на торговых ярмарках, или автоматические экскурсоводы в музеях. Для успешного выполнения задач обслуживания требуется взаимодействие с людьми и способность надежно реагировать на непредсказуемые и динамические изменения в среде.
- **Развлечения.** Роботы начали завоевывать индустрию игр и развлечений. Выше уже шла речь о роботе AIBO компании Sony (см. рис. 25.4, б); этот игрушечный робот, похожий на собаку, используется в качестве исследовательской

платформы в лабораториях искусственного интеллекта во всем мире. Одной из самых интересных задач искусственного интеллекта, которые изучаются с помощью этой платформы, является **робототехнический футбол**, — соревновательная игра, весьма напоминающая тот футбол, в который играют люди, но проводимая с участием автономных мобильных роботов. Робототехнический футбол открывает широкие возможности для исследований по искусственному интеллекту, поскольку ставит в повестку дня целый ряд задач, способных служить прототипами для многих других, более серьезных робототехнических приложений. Ежегодные робототехнические футбольные соревнования привлекают интерес многих исследователей по искусственному интеллекту и вносят много приятных моментов в эту область робототехники.

- **Дополнение возможностей людей.** Последней прикладной областью робототехнической технологии является дополнение возможностей людей. Например, исследователи разработали шагающие машины для прогулок, которые могут использоваться людьми для передвижения вместо инвалидных колясок. Кроме того, в некоторых исследованиях основные усилия в настоящее время сосредоточены на разработке устройств, которые упрощают для людей задачу ходьбы или передвижения их рук путем приложения дополнительных усилий, действующих через устройства, закрепленные вне скелета. Если крепления таких устройств установлены на постоянной основе, то их можно рассматривать как искусственные роботизированные конечности. Еще одной формой дополнения возможностей людей является роботизированное дистанционное выполнение операций, или так называемое *телеприсутствие*. Дистанционное выполнение операций — это осуществление рабочих заданий на больших расстояниях с помощью робототехнических устройств. При выполнении роботизированных дистанционных операций широко применяется такая конфигурация, как “ведущий—ведомый”, в которой робот-манипулятор повторяет движения действующего на удалении оператора-человека, которые передаются через тактильный интерфейс. Все эти системы дополняют возможности человека по взаимодействию с его средой. Некоторые проекты заходят столь далеко, что в них предпринимается попытка создать некое подобие человека, по меньшей мере на очень поверхностном уровне. Например, в настоящее время некоторые японские компании выполняют коммерческие поставки роботов-гуманоидов.

25.9. РЕЗЮМЕ

Робототехника — это научно-техническое направление, посвященное созданию интеллектуальных агентов, которые выполняют свои манипуляции в физическом мире. В данной главе представлены перечисленные ниже основные сведения об аппаратном и программном обеспечении роботов.

- Роботы оснащены датчиками для получения результатов восприятия из своей среды и исполнительными механизмами, с помощью которых они могут прилагать физические усилия в своей среде. Большинство роботов представляют

собой либо манипуляторы, закрепленные в фиксированных позициях, либо мобильные роботы, способные передвигаться.

- Робототехническое восприятие предназначено для оценки количественных значений, необходимых для принятия решений, на основании сенсорных данных. Для выполнения этих функций требуются внутреннее представление и метод обновления этого внутреннего представления во времени. В число широко известных сложных задач восприятия входят локализация и составление карты.
- В процессе робототехнического восприятия могут применяться такие вероятностные алгоритмы фильтрации, как фильтры Калмана и фильтры частиц. Эти методы обеспечивают поддержку доверительного состояния, т.е. распределение апостериорных вероятностей по переменным состояниям.
- Планирование движений робота обычно осуществляется в пространстве конфигураций, каждая точка которого определяет местонахождение и ориентацию робота, а также углы поворота его шарниров.
- В состав алгоритмов поиска в пространстве конфигураций входят методы декомпозиции ячеек, которые предусматривают декомпозицию пространства всех конфигураций на бесконечное количество ячеек, и методы скелетирования, предусматривающие создание проекций пространства конфигураций на пространства с меньшими размерностями. После этого задача планирования движений решается с использованием поиска в этих более простых структурах.
- Задачу прохождения по пути, найденному с помощью алгоритма поиска, можно решить, используя этот путь в качестве опорной траектории для PID-контроллеров.
- При использовании методов на основе поля потенциалов навигация роботов осуществляется с применением функций потенциалов, параметрами которых служат расстояния до препятствий и целевое местонахождение. Недостатком методов на основе поля потенциалов является то, что при их использовании могут возникать тупиковые ситуации, при которых робот не может выйти из локального минимума. Тем не менее эти методы позволяют непосредственно вырабатывать команды на осуществление движения, не требуя планирования пути.
- Иногда проще непосредственно задать спецификацию контроллера робота, а не определять алгоритмическим способом путь с помощью явно заданной модели среды. Такие контроллеры часто могут быть выполнены в виде простых конечных автоматов.
- Обобщающая архитектура позволяет программистам собирать контроллеры роботов из взаимосвязанных конечных автоматов, дополненных встроенными таймерами.
- Для разработки робототехнического программного обеспечения, в котором объединяются средства алгоритмического вывода, упорядочения подцелей и управления, широко применяется инфраструктура, основанная на трехуровневой архитектуре.
- Для упрощения разработки программного обеспечения роботов используются языки программирования специального назначения. В этих языках предусмотрены конструкции, предназначенные для разработки многопоточкового

программного обеспечения, для интеграции директив управления в средства планирования и для обучения на основе опыта.

БИБЛИОГРАФИЧЕСКИЕ И ИСТОРИЧЕСКИЕ ЗАМЕТКИ

Слово **робот** получило широкую известность после выхода в 1921 году пьесы R.U.R. (Rossum's Universal Robots — универсальные роботы Россума) чешского драматурга Карела Чапека. В этой пьесе роботы, которые выращивались из химических растворов, а не конструировались механически, в конечном итоге взбунтовались против своих создателей и решили взять над ними верх. Считается [562], что это слово фактически придумал брат Карела Чапека, Йозеф, который скомбинировал чешские слова “robota” (принудительный труд) и “robotnik” (раб), получив слово “robot”, которое он применил в своем рассказе *Opilec*, изданном в 1917 году.

Термин “робототехника” (robotics) был впервые использован Айзеком Азимовым [46]. Но само понятие робототехники (которое было известно под многими другими названиями) имеет гораздо более долгую историю. В древнегреческой мифологии упоминается механический человек по имени Талос, который был спроектирован и построен греческим богом огня и кузнечного дела Гефестом. В XVIII столетии разрабатывались восхитительные автоматы (одним из первых примеров таких автоматов является механическая утка Жака Вокансона, созданная в 1738 году), но сложное поведение, которое они демонстрировали, было полностью задано заранее. Возможно, одним из примеров программируемого устройства, подобного роботу, был ткацкий станок Жаккарда (1805 год), описанный в главе 1.

Первым коммерчески поставляемым роботом был робот-манипулятор, получивший название **Unimate** (сокращение от universal automation — универсальная автоматизация). Робот Unimate был разработан Джозефом Энджелбергером и Джорджем Деволом. В 1961 году первый робот Unimate был продан компании General Motors, в которой он использовался при изготовлении телевизионных трубок. Тот же 1961 год примечателен и тем, что в этом году Девол получил первый патент США на конструкцию робота. Через одиннадцать лет, в 1972 году, корпорация Nissan впервые автоматизировала весь сборочный конвейер с помощью роботов, разработанных компанией Kawasaki на основе роботов, поставляемых компанией Unimation Энджелбергера и Девол. Эта разработка стала началом важной научно-технической революции, которая вначале происходила в основном в Японии и США и до сих пор продолжается во всем мире. Следующий шаг был сделан компанией Unimation в 1978 году, когда ее сотрудниками был разработан робот **PUMA** (сокращение от Programmable Universal Machine for Assembly — программируемая универсальная машина для сборки). Робот PUMA, первоначально разработанный для компании General Motors, стал фактически стандартом робототехнических манипуляторов на два следующих десятилетия. В настоящее время количество действующих роботов во всем мире оценивается в один миллион, причем больше половины из них установлено в Японии.

Литературу по робототехническим исследованиям можно разделить приблизительно на две части: мобильные роботы и стационарные манипуляторы. Первым автономным мобильным роботом можно считать “черепаху” Грея Уолтера, которая была создана в 1948 году, хотя ее система управления не была программируемой.

Робот “Зверь Хопкинса” (Hopkins Beast), созданный в начале 1960-х годов в Университете Джона Хопкинса, был гораздо более сложным; он имел аппаратные средства распознавания образов и был способен обнаружить крышку стандартной розетки питания переменным током. Этот робот обладал способностью находить розетки, подключаться к ним и перезаряжать свои аккумуляторы! Тем не менее “Зверь” имел ограниченный набор навыков. Первым мобильным роботом общего назначения был “Shakey”, разработанный в конце 1990-х годов в институте, называвшемся тогда Станфордским научно-исследовательским институтом (Stanford Research Institute; теперь он носит название SRI) [466], [1143]. “Shakey” был первым роботом, в котором объединялись функции восприятия, планирования и выполнения, и это замечательное достижение оказало влияние на многие дальнейшие исследования по искусственному интеллекту. К другим важным проектам относятся Stanford Cart и CMU Rover [1082]. В [303] описано классическое исследование по автономным транспортным средствам.

В основе развития области робототехнической картографии лежат два различных источника. Первое направление начиналось с работы Смита и Чизмена [1440], которые применяли фильтры Калмана для одновременного решения задач локализации и составления карты. Разработанный ими алгоритм был впервые реализован в исследовании, описанном в [1095], а в дальнейшем дополнен в [912]. В [400] описано современное состояние дел в этой области. Второе направление началось с разработки представления в виде **сетки занятости** для вероятностной картографии, в которой задается вероятность того, что каждый участок (x, y) занят некоторым препятствием [1083]. Обзор современного состояния дел в робототехнической картографии можно найти в [1508]. Одной из первых работ, в которой было предложено использовать топологическую, а не метрическую картографию, стимулом для которой послужили модели пространственного познания человека, была [863].

Обзор самых ранних методов локализации мобильных роботов приведен в [153]. В теории управления фильтры Калмана применяются в качестве метода локализации в течение многих десятилетий, а в литературе по искусственному интеллекту общая вероятностная формулировка задачи локализации появилась гораздо позже, благодаря работам Тома Дина и его коллег [356], [362], а также Симонса и Кёнига [1412]. В последней работе был предложен термин **марковская локализация**. Первым практическим применением этого метода явилась работа Бургарда и др. [208], которые создали целый ряд роботов, предназначенных для использования в музеях. Метод локализации Монте-Карло, основанный на фильтрах частиц, был разработан Фоксом и др. [488] и в настоящее время получил широкое распространение. В **фильтре частиц, действующем по принципу Рао-Блэквелла**, объединяются средства фильтрации частиц для локализации робота со средствами точной фильтрации для составления карты [1074], [1106].

В течение последнего десятилетия общим стимулом для проведения исследований по мобильным роботам служат два важных соревнования. Ежегодное соревнование мобильных роботов общества AAAI впервые было проведено в 1992 году. Первым победителем соревнования стал робот Carmel [287]. Наблюдаемый прогресс остается стабильным и весьма впечатляющим: в одном из последних соревнований (2002 год) перед роботами стояла задача войти в комплекс зданий, где проводилась конференция, найти путь к бюро регистрации, зарегистрироваться на конференции и произнести речь. В соревновании **Robocup**, инициатором проведения которого в 1995 году стал Китано со своими коллегами [802], провозглашена цель — к 2050 году

“разработать команду полностью автономных роботов-гуманоидов, которые смогут победить команду чемпионов мира по футболу среди людей”. Игры происходят в лигах для роботов, имитирующих людей, колесных роботов различных размеров и четырехногих роботов Aibo компании Sony. В 2002 году это соревнование привлекло команды почти из 30 различных стран и на нем присутствовало больше 100 тысяч зрителей.

Исследования по созданию роботов-манипуляторов, которые первоначально именовались **машинами “рука—глаз”**, развиваются в нескольких весьма различных направлениях. Первой важной работой по созданию машины “рука—глаз” был проект Генриха Эрнста под названием МН-1, описанный в тезисах его докторской диссертации, которые были опубликованы в Массачусеттском технологическом институте [441]. Проект Machine Intelligence, разработанный в Эдинбурге, также стал одной из первых демонстраций впечатляющей сборочной системы, основанной на использовании средств машинного зрения, которая получила название Freddy [1044]. После того как был сделан этот решающий начальный вклад, основные усилия были сосредоточены на создании геометрических алгоритмов для решения задач планирования движения в детерминированной и полностью наблюдаемой среде. В оригинальной работе Рейфа [1274] было показано, что задача планирования движения робота является PSPACE-трудной. Представление на основе пространства конфигураций было предложено Лозано-Пересом [956]. Важное влияние оказал ряд статей Шварца и Шарира, посвященных задачам, которые были ими названы задачами для **грузчиков-тяжеловесов** (piano mover) [1371].

Метод рекурсивной декомпозиции ячеек для планирования в пространстве конфигураций был впервые предложен в [193] и существенно доработан в [1646]. Самые первые алгоритмы скелетирования были основаны на диаграммах Вороного [1313] и **графах видимости** [1581]. В [603] приведены результаты разработки эффективных методов инкрементного вычисления диаграмм Вороного, а в [253] диаграммы Вороного были распространены на гораздо более широкий класс задач планирования движения. В тезисах докторской диссертации Джона Кэнни [219] предложен первый полностью экспоненциальный алгоритм планирования движения с использованием еще одного метода скелетирования, называемого алгоритмом **силуэта**. В книге Жан-Клода Латомба [891] рассматривается целый ряд подходов к решению задачи планирования движения. В [780] описаны методы разработки вероятностных дорожных карт, которые в настоящее время относятся к числу наиболее эффективных методов. Планирование тонких движений с ограниченными сенсорными данными исследовалось в [217] и [957] на основе идеи интервальной неопределенности, а не вероятностной неопределенности. В навигации на основе отметок [902] используются во многом такие же идеи, как и в других областях исследования мобильных роботов.

Управлению роботами как динамическими системами (для решения задач манипулирования или навигации) посвящен огромный объем литературы, поэтому в данной главе лишь кратко рассматриваются сведения, касающиеся этой области. К числу важнейших работ относятся трехтомник по импедансному управлению Хогана [668] и общее исследование по динамике роботов Физерстоуна [456]. Дин и Уэллман [363] были в числе первых, кто попытался связать воедино теорию управления и системы планирования на основе искусственного интеллекта. Три классическими учебниками по математическим основам робототехнического манипулирования являются [305], [1184] и [1634]. Важное значение в робототехнике имеет также область проблем **схватывания**, поскольку тема определения стабильных ус-

ловий приложения захвата является весьма сложной [998]. Для того чтобы иметь возможность создать успешно действующее средство схватывания, необходимо обеспечить восприятие данных о прикосновении, или ~~тактильную обратную связь~~, для определения усилия контакта и обнаружения проскальзывания [455].

Понятие управления на основе поля потенциалов, с помощью которого предпринимаются попытки одновременно решать задачи планирования движения и управления, было введено в сферу робототехники Хатибом [793]. В технологии мобильных роботов эта идея стала рассматриваться как практический способ решения задачи предотвращения столкновений, а в дальнейшем была дополнена и легла в основу алгоритма ~~гистограмм векторного поля~~ [154]. Функции навигации, представляющие собой робототехническую версию политики управления для детерминированных задач MDP, были предложены в [812].

Вокруг проблемы создания архитектур программного обеспечения для роботов ведутся оживленные дискуссии. Истоками трехуровневой архитектуры стали исследования, проводившиеся еще в эпоху так называемого “старого доброго искусственного интеллекта”, в том числе исследования по созданию проекта Shakey; общий обзор на эту тему приведен в [525]. Обобщающая архитектура была предложена Родни Бруксом [189], хотя аналогичные идеи были разработаны независимо Брейтенбергом [172], в книге которого, *Vehicles*, описан ряд простых роботов, основанных на поведенческом подходе. Вслед за успешным созданием Бруксом шестиногого шагающего робота был разработан целый ряд других проектов. Коннелл описал в своих тезисах докторской диссертации [288] результаты разработки мобильного робота, способного находить объекты, проект которого был полностью реактивным. Описания попыток распространить поведенческий принцип на системы, состоящие из нескольких роботов, можно найти в [999] и [1176]. Результатом обобщения идей робототехники, основанной на параллельно организуемом поведении, стало создание универсальных языков управления роботами GRL [681] и Colbert [832]. В [38] приведен обзор состояния дел в этой области.

Для управления мобильными роботами, выполняющими задания по разведке местности и доставке материалов, использовались также **ситуационные автоматы** [760], [1308], описанные в главе 7. Проекты ситуационных автоматов тесно связаны с проектами, основанными на организации поведения, поскольку они состоят из конечных автоматов, отслеживающих различные аспекты состояния среды с использованием простых комбинаторных схем. Тем не менее в подходе, основанном на организации поведения, подчеркивается отсутствие явного представления, а ситуационные автоматы конструируются алгоритмически из декларативных моделей среды таким образом, что представительное содержание каждого регистра состояния является полностью определенным.

В настоящее время имеется несколько хороших современных учебников по мобильной робототехнике. Кроме учебников, указанных выше, можно назвать сборник Кортенкампа и др. [846], в котором приведен исчерпывающий обзор современных архитектур систем мобильных роботов. В недавно изданных учебниках [423] и [1107] рассматриваются более общие вопросы робототехники. Еще в одной недавно изданной книге по робототехническому манипулированию рассматриваются такие сложные темы, как согласующее движение [997]. Основной конференцией по робототехнике является *IEEE International Conference on Robotics and Automation*. В число робо-

тотехнических журналов входят *IEEE Robotics and Automation*, *International Journal of Robotics Research* и *Robotics and Autonomous Systems*.

УПРАЖНЕНИЯ

25.1. При любом конечном размере выборки результаты применения алгоритма локализации Монте-Карло являются смещенными (т.е. ожидаемое значение данных о местонахождении, вычисленные с помощью алгоритма, отличаются от истинного ожидаемого значения), поскольку именно так действуют алгоритм фильтрации частиц. В данном упражнении предлагается оценить это смещение.

Для упрощения рассмотрим мир с четырьмя возможными местонахождениями робота: $X = \{x_1, x_2, x_3, x_4\}$. Первоначально осуществим равномерную выборку $N \geq 1$ образцов среди этих местонахождений. Как обычно, вполне приемлемо, если для любого из местонахождений x будет сформировано больше одной выборки. Допустим, что Z — булева сенсорная переменная, характеризующаяся следующими условными вероятностями:

$$P(z|x_1) = 0.8 \quad P(\neg z|x_1) = 0.2$$

$$P(z|x_2) = 0.4 \quad P(\neg z|x_2) = 0.6$$

$$P(z|x_3) = 0.1 \quad P(\neg z|x_3) = 0.9$$

$$P(z|x_4) = 0.1 \quad P(\neg z|x_4) = 0.9$$



В алгоритме MCL эти вероятности используются для формирования весов частиц, которые в дальнейшем нормализуются и используются в процессе повторной выборки. Для упрощения предположим, что при повторной выборке вырабатывается только один новый образец, независимо от N . Этот образец может соответствовать любому из четырех местонахождений x . Таким образом, процесс выборки определяет распределение вероятностей по x .

- а) Каково результирующее распределение вероятностей по x для этого нового образца? Ответьте на этот вопрос отдельно для $N=1, \dots, 10$ и для $N=\infty$.
- б) Разница между двумя распределениями вероятностей P и Q может быть измерена с помощью дивергенции KL , которая определяется следующим образом:

$$KL(P, Q) = \sum_i P(x_i) \log \frac{P(x_i)}{Q(x_i)}$$

Каковы значения дивергенции KL между распределениями в упр. 25.1, а и истинным распределением апостериорных вероятностей?

- в) Какая модификация формулировки задачи (а не алгоритма!) гарантировала бы, чтобы конкретное приведенное выше выражение для оценки оставалось несмещенным даже при конечных значениях N ? Предложите по меньшей мере две такие модификации (но каждая из них должна соответствовать предложенному заданию).

- 25.2.  Реализуйте алгоритм локализации Монте-Карло для моделируемого робота с датчиками расстояний. Карту, нанесенную на сетку, и данные о расстояниях можно найти в репозитории кода по адресу `aima.cs.berkeley.edu`. Упражнение будет считаться выполненным, если вы продемонстрируете успешную глобальную локализацию робота.
- 25.3. Рассмотрим робот-манипулятор, показанный на рис. 25.11. Предположим, что элемент у основания робота имеет длину 60 см, а плечо и предплечье имеют длину по 40 см. Как было указано на с. 1, обратная кинематика робота часто является не уникальной. Сформулируйте явное решение в замкнутой форме для обратной кинематики этого манипулятора. При каких именно условиях это решение является уникальным?
- 25.4.  Реализуйте алгоритм вычисления диаграммы Вороного для произвольной двумерной среды, описанной с помощью булева массива с размерами $n \times n$. Проиллюстрируйте работу вашего алгоритма, построив график диаграммы Вороного для 10 интересных карт. Какова сложность вашего алгоритма?
- 25.5. В этом упражнении рассматривается связь между рабочим пространством и пространством конфигураций с использованием примеров, показанных на рис. 25.24.

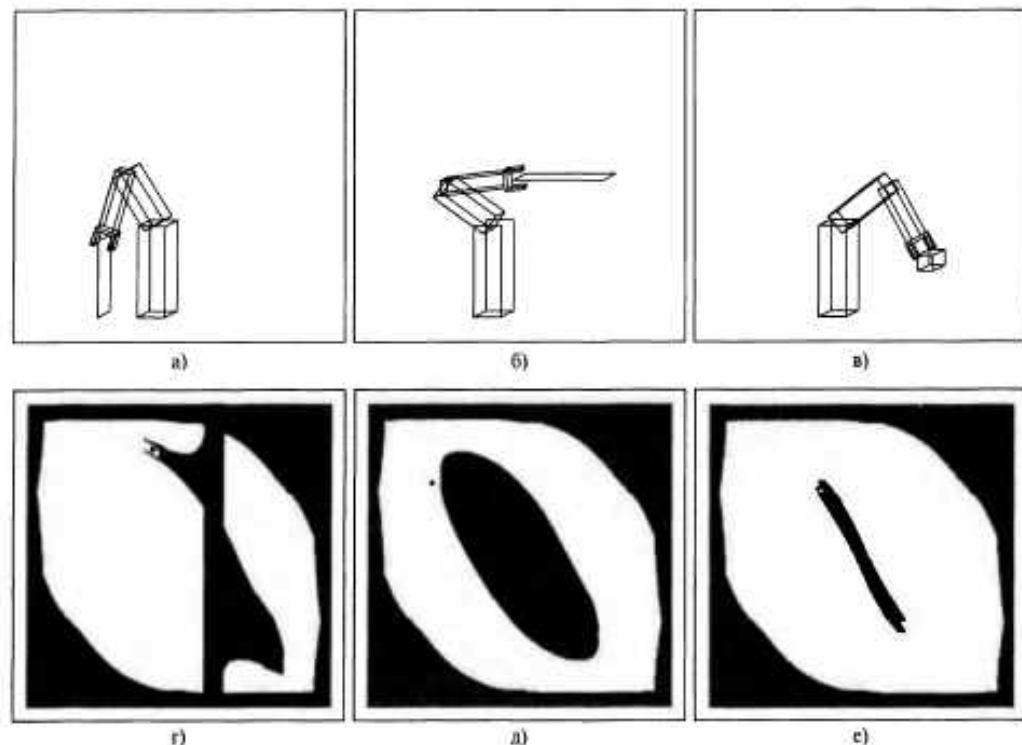


Рис. 25.24. Схемы для упр. 25.5

- а) Рассмотрим конфигурации роботов, показанных на рис. 25.24, а–в, игнорируя препятствие, приведенное на каждом из этих рисунков. Нари-

суйте соответствующие конфигурации манипулятора в пространстве конфигураций. (*Подсказка.* Каждая конфигурация манипулятора отображается на единственную точку в пространстве конфигураций, как показано на рис. 25.11, б.)

- б) Нарисуйте пространство конфигураций для каждой из диаграмм рабочего пространства, приведенных на рис. 25.24, а–в. (*Подсказка.* В этих пространствах конфигураций общим с пространством конфигураций, показанным на рис. 25.24, а, является тот участок, который соответствует столкновению манипулятора робота с самим собой, а различия обусловлены отсутствием окружающих препятствий и изменением местонахождений препятствий на этих отдельных рисунках.)
- в) Для каждой из черных точек, приведенных на рис. 25.24, д, е, нарисуйте соответствующие конфигурации манипулятора робота в рабочем пространстве. В этом упражнении не рассматривайте затененные участки.
- г) Все пространства конфигураций, показанные на рис. 25.24, д, е, сформированы с учетом единственного препятствия в рабочем пространстве (темное затенение), а также ограничений, обусловленных ограничением, препятствующим столкновению манипулятора с самим собой (светлое затенение). Для каждой диаграммы нарисуйте препятствие в рабочем пространстве, которое соответствует участку с темным затенением.
- д) На рис. 25.24, з, показано, что единственное плоское препятствие способно разбить рабочее пространство на два несвязанных между собой участка. Каково максимальное количество несвязанных участков, которые могут быть созданы в результате вставки плоского препятствия в свободное от препятствий связное рабочее пространство для робота с двумя степенями свободы? Приведите пример и объясните, почему не может быть создано большее количество несвязных участков. Относится ли это утверждение к неплоскому препятствию?

25.6. Рассмотрим упрощенный робот, показанный на рис. 25.25. Предположим, что декартовы координаты робота всегда известны, а также известны координаты его целевого местонахождения. Тем не менее неизвестны местонахождения препятствий. Робот, как показано на этом рисунке, может обнаруживать препятствия, находящиеся в непосредственной близости от него. Для упрощения предположим, что движения робота не подвержены шуму и что пространство состояний является дискретным. На рис. 25.25 приведен только один пример; в этом упражнении требуется найти решение для всех возможных миров, заданных в координатной сетке, где действительно имеется путь от начала до целевого местонахождения.

- а) Спроектируйте алгоритмический контроллер, который гарантирует, что робот всегда достигнет своего целевого местонахождения, если это вообще возможно. Этот алгоритмический контроллер может запоминать в форме карты результаты измерений, которые он получает по мере передвижения робота. Между отдельными операциями перемещения робот может затрачивать произвольно долгое время на алгоритмическую обработку информации.

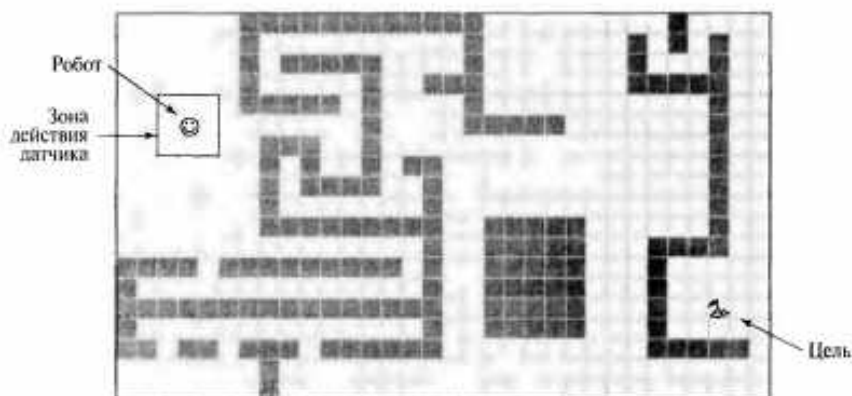


Рис. 25.25. Упрощенный робот в лабиринте (см. упр. 25.6)

- б) Теперь спроектируйте реактивный контроллер для выполнения той же задачи. Этот контроллер может не запоминать полученные ранее результаты сенсорных измерения. (Он может даже не составлять карту!) Вместо этого он должен принимать все решения на основании текущих результатов измерений, которые включают данные о его собственном местонахождении и о местонахождении цели. Время, необходимое для выработки решения, не должно зависеть от размеров среды или от количества ранее выполненных шагов. Каково максимальное количество шагов, которые могут потребоваться роботу для достижения цели?
- в) Какую производительность покажут контроллеры, созданные по условиям упр. 25.6, а и б, если учитываются только следующие шесть условий: непрерывное пространство состояний, зашумленные результаты восприятия, зашумленные результаты движения, зашумленные результаты и восприятия, и движения, неизвестное местонахождение цели (цель может быть обнаружена только в пределах действия датчика расстояний) или движущиеся препятствия. Для каждого условия и каждого контроллера приведите пример ситуации, в которой робот не достигает цели (или объясните, почему такая ситуация не может возникнуть).

- 25.7. На рис. 25.21, б показан дополненный конечный автомат для управления одной ногой шестиногого робота. Цель этого упражнения состоит в том, чтобы спроектировать автомат AFSM, который объединяет шесть экземпляров отдельных контроллеров ног и обеспечивает эффективное, стабильное движение. Для этой цели вы должны дополнить контроллер отдельной ноги так, чтобы он передавал сообщения вновь разрабатываемому автомату AFSM и ожидал поступления других сообщений. Докажите, почему предложенный вами контроллер является эффективным, в том отношении, что он не затрачивает энергию непроизводительно (например, на приведение в движение ног, проскальзывающих в каком-то месте) и что он продвигает робота вперед с достаточно высокой скоростью. Докажите, что предложенный вами контроллер удовлетворяет условию стабильности, приведенному на с. 1.
- 25.8. (Это упражнение было впервые предложено Майклом Генезеретом (Michael Genesereth) и Нильсом Нильссоном (Nils Nilsson). В нем могут участвовать

и студенты-первокурсники, и аспиранты.) Люди так успешно выполняют простейшие задачи даже без помощи пальцев, например берут со стола чашки или укладывают кубики в столбики, что часто не осознают, насколько сложными являются эти задачи. Данное упражнение позволяет раскрыть всю сложность элементарных задач и снова пройти путь развития робототехники за последние 30 лет. Вначале нужно выбрать задачу, такую как составление арки из трех блоков. Затем необходимо организовать работу робота с использованием четырех людей, как описано ниже.

- Мозг. Задача Мозга состоит в том, что он должен составлять план достижения цели и управлять руками при выполнении этого плана. Мозг получает входные данные от Глаз, но не может видеть непосредственно саму сцену. Мозг является единственным, кто знает, в чем состоит цель.
- Глаза. Задача Глаз состоит в том, чтобы сообщать Мозгу краткое описание сцены. Глаза должны находиться на расстоянии одного-двух метров от рабочей среды и могут предоставлять ее качественное описание (например, “красная коробка стоит на зеленой коробке, лежащей на боку”) или количественное описание (“зеленая коробка находится слева от синего цилиндра, на расстоянии около полуметра”). Глаза могут также отвечать примерно на такие вопросы Мозга: “Есть ли промежуток междулевой Рукой и красной коробкой?” Если в вашем распоряжении имеется видеокамера, направьте ее на сцену и разрешите Глазам смотреть в видеоискатель видеокамеры, а не прямо на сцену.
- Левая Рука и Правая Рука. Роль каждой Руки играют по одному человеку. Две Руки стоят рядом друг с другом; Левая Рука использует только свою левую руку, а Правая Рука — только свою правую руку. Руки выполняют лишь простые команды от Мозга, например: “Левая Рука, передвинься на пять сантиметров вперед”. Руки не могут выполнять команды, отличные от движений; например: “Подними коробку” — это не та команда, которую может выполнить Рука. Для предотвращения попыток действовать пальцами можно предусмотреть, чтобы Руки носили рукавицы или действовали с помощью клещей. Глаза у Рук должны быть завязаны. Единственные сенсорные возможности, которые им предоставляются, таковы: они имеют право сообщить о том, что путь их движения заблокирован неподвижным препятствием, таким как стол или другая Рука. В подобных случаях Руки могут лишь подать звуковой сигнал, чтобы сообщить Мозгу о возникшем затруднении.

Часть VIII

ЗАКЛЮЧЕНИЕ

Философские основания	1248
Настоящее и будущее искусственного интеллекта	1259

26 ФИЛОСОФСКИЕ ОСНОВАНИЯ

В данной главе речь идет о том, что означает “мыслить”, а также о том, могут ли и должны ли этим заниматься искусственно созданные объекты.

Как упоминалось в главе 1, философы размышляли над мировыми проблемами задолго до того, как появились компьютеры, и пытались решить некоторые проблемы, которые по сути относятся к искусственному интеллекту: “Как функционирует разум? Возможно ли, чтобы машины действовали столь же интеллектуально, как люди, а если ответ на этот вопрос является положительным, то будет ли это означать, что они обладают разумом? Каковы этические последствия создания интеллектуальных машин?” Во всех предыдущих главах настоящей книги рассматривались вопросы, касающиеся самого искусственного интеллекта, а здесь мы в рамках одной главы рассмотрим указанные выше философские проблемы.

Прежде всего введем некоторую терминологию: утверждение, согласно которому машины, возможно, обладают способностью действовать интеллектуально (по-видимому, эту мысль лучше выразить таким образом, что машины, возможно, способны действовать так, как будто действительно являются интеллектуальными), философы называют гипотезой **слабого искусственного интеллекта**, а утверждение, что машины действительно мыслят (а не просто имитируют мыслительные процессы), называется гипотезой **сильного искусственного интеллекта**.

Большинство исследователей искусственного интеллекта принимают гипотезу слабого искусственного интеллекта как данную и не задумываются над тем, что может рассматриваться также гипотеза сильного искусственного интеллекта; коль скоро разработанная ими программа успешно функционирует, их не волнует, назовут ли ее работу имитацией интеллекта или настоящим интеллектом. Но всех исследователей искусственного интеллекта должны заботить этические последствия их деятельности.

26.1. СЛАБЫЙ ИСКУССТВЕННЫЙ ИНТЕЛЛЕКТ: МОГУТ ЛИ МАШИНЫ ДЕЙСТВОВАТЬ ИНТЕЛЛЕКТУАЛЬНО?

Некоторые философы пытались доказать, что создать искусственный интеллект невозможно, т.е. что машины никогда не смогут действовать интеллектуально. Некоторые из них даже использовали свою эрудицию, чтобы призвать всех прекратить исследования искусственного интеллекта, приводя следующие доводы.

Искусственный интеллект, создаваемый в рамках культа компьютероцентризма, не дает даже ни малейшего шанса на то, что с его помощью удастся добиться долговременных результатов ... настало время направить усилия исследователей искусственного интеллекта (и значительные средства, выделяемые на их поддержку) в области, отличные от этого компьютеризированного подхода [1355].

Очевидно, что ответ на вопрос о том, возможно или невозможно создание искусственного интеллекта, зависит от того, как определено само понятие искусственного интеллекта. По существу создание искусственного интеллекта — это борьба за разработку наилучшей возможной программы агента в данной конкретной архитектуре. При использовании такой формулировки создание искусственного интеллекта возможно по определению, поскольку для любой цифровой архитектуры, состоящей из k битов памяти, существует точно 2^k программ агентов, и для того чтобы найти наилучшую из них, достаточно просто последовательно проверить их все. Такой подход может оказаться неосуществимым при больших значениях k , но философы оперируют с теоретическими, а не практическими конструкциями.

Приведенное выше определение искусственного интеллекта вполне подходит для решения технической проблемы поиска приемлемой программы агента при наличии некоторой заданной архитектуры. Поэтому вполне можно было бы закончить этот раздел прямо сейчас, ответив положительно на вопрос, приведенный в его названии. Но философов интересует также проблема сравнения двух вычислительных архитектур — человека и машины. К тому же в философии существует давняя традиция поиска ответа на вопрос: **Могут ли машины мыслить?** К сожалению, сам этот вопрос определен неправильно. Для того чтобы понять, почему это так, рассмотрим приведенные ниже вопросы.

- Могут ли машины совершать полеты?
- Могут ли машины заниматься плаванием?

Большинство людей согласятся, что ответ на первый вопрос является положительным, поскольку самолеты предназначены именно для того, чтобы совершать полеты, но ответят на второй вопрос отрицательно; корабли и подводные лодки движутся над водой и под водой, но это мы не называем *занятиями плаванием*. Тем не менее ни приведенные вопросы, ни ответы не оказывают никакого влияния на повседневную деятельность специалистов по авионавигации и судоходству, а также на пользователей тех машин, которыми они управляют. К тому же ответы на эти вопросы никак не могут помочь лучше проектировать или расширять возможности самолетов и подводных лодок и в большей степени касаются выбора правильных способов словоупотребления. Слово “заниматься плаванием” (swim) в английском языке постепенно приняло смысл “продвигаться в воде за счет движений частей тела”, а слово “совершать полеты” (fly) не налагает таких ограничений на выбор способов

передвижения¹. Практическая возможность пользоваться услугами “мыслящих машин” предоставляется человеку в течение всего лишь 50 лет, а этого еще недостаточно для того, чтобы толкование слова “мыслить” распространилось и на машины.

Алан Тьюринг в своей знаменитой статье “*Computing Machinery and Intelligence*” [1520] указал, что вместо поиска ответа на вопрос, могут ли машины мыслить, мы должны интересоваться тем, могут ли машины пройти поведенческий тест интеллектуальности, который в дальнейшем получил название *теста Тьюринга*. Этот тест заключается в том, что программа в течение 5 минут участвует в разговоре (складывающемся из сообщений, которые передаются в оперативном режиме) с некоторым собеседником. Затем этот собеседник должен определить, проводился ли этот разговор с программой или с другим человеком; программа успешно проходит тест, если ей удастся обмануть собеседника в 30% случаев. Тьюринг предсказал, что к 2000 году компьютер с объемом памяти в 10^9 единиц удастся запрограммировать настолько успешно, чтобы он прошел этот тест, но оказался неправ. Некоторых людей еще раньше удавалось водить за нос в течение 5 минут; например, программа Eliza и чатбот (робот-собеседник) Mgonz, действующий в Internet, не раз обманывали людей, которые так и не могли понять, что они общаются с программой, а программа Alice смогла даже одурачить судью на соревнованиях за приз Лёбнера (Loebner Prize) 2001 года. Но ни одной программе не удалось приблизиться к 30%-ному критерию в борьбе против специально обученных судей, да и в самой области искусственного интеллекта тесту Тьюринга уделяют все меньше внимания.

Тьюринг также исследовал широкий перечень вероятных возражений против самой возможности создания интеллектуальных машин, в том числе сумел предвидеть практически все возражения, которые были выдвинуты в течение полувека, прошедшего со времени появления его статьи. В этой главе будут рассмотрены некоторые из них.

Довод, исходящий из неспособности

В “доводе, исходящем из неспособности”, выдвигается претензия, что “машина никогда не сможет выполнить действие X”. В качестве примеров действий X Тьюринг приводит следующий список.

Быть добрым, щедрым, красивым, дружелюбным, инициативным, иметь чувство юмора, отличать правду от лжи, совершать ошибки, влюбляться, наслаждаться земляникой и мороженым, заставлять влюбляться в себя, учиться на опыте, правильно употреблять слова, быть предметом собственных мыслей, проявлять такое же разнообразие в поведении, как и человек, создавать действительно что-то новое.

Тьюрингу пришлось воспользоваться своей интуицией для выдвижения предположений о том, что будет возможно в будущем, а мы имеем счастливую возможность оглянуться назад, чтобы узнать, чего уже удалось добиться компьютерам. Нельзя отрицать, что компьютеры в наши дни выполняют многие действия, которые раньше были прерогативой одних лишь людей. Программы играют в шахматы, шашки и другие игры, контролируют детали на сборочных конвейерах, проверяют правописание в документах, введенных с помощью текстового редактора, водят автомобили

¹ В русском языке эквивалент английского слова “swim” не носит такого ограничительного оттенка, поэтому может применяться и к судам.

и вертолеты, диагностируют заболевания, а также выполняют сотни других работ столь же хорошо, как люди, или даже лучше. Компьютеры сделали небольшие, но важные открытия в астрономии, математике, химии, минералогии, биологии, компьютерных науках и других областях. В каждом из этих случаев требовалось обладать способностями на уровне человека-эксперта.

С учетом того что мы знаем о компьютерах, не удивительно, что они легко справляются с такими комбинаторными задачами, как игра в шахматы. Но алгоритмы действуют также на уровнях не ниже человека при решении таких задач, которые, на первый взгляд, не позволяют обойтись без человеческого суждения или, как выразил это Тьюринг, заставляют “учиться на опыте” и требуют наличия способностей “отличать правду от лжи”. Еще в 1955 году Пауль Мель [1032] (см. также [600]) изучал процессы принятия решений обученными экспертами как субъективные задачи, аналогичные прогнозированию успешного овладения студентом программы обучения или повторного совершения преступления рецидивистом. В 19 из 20 рассмотренных им исследований Мель обнаружил, что простые алгоритмы статистического обучения (такие как алгоритмы линейной регрессии или наивные байесовские алгоритмы) вырабатывают лучшие предсказания, чем люди-эксперты. В американской Службе образовательного тестирования (Educational Testing Service) с 1999 года использовалась автоматизированная программа для выставления оценок по обзорным вопросам на экзаменах GMAT. Результаты работы программы согласовывались с результатами работы экзаменаторов, которым было поручено выставять такие оценки, в 97% случаев, а этот уровень соответствует совпадению оценок двух экзаменаторов [212].

Очевидно, что компьютеры могут выполнять многие действия столь же успешно или даже лучше, чем люди, включая такие работы, в которых, по мнению людей, требуется огромная человеческая прозорливость и понимание. Это, безусловно, не означает, что компьютеры при выполнении этих работ проявляют прозорливость и понимание (указанные свойства не входят в состав поведения, о чем будет сказано ниже), но суть заключается в том, что первые предположения о содержании мыслительных процессов, требуемых для выработки данного конкретного поведения, часто оказываются ложными. Безусловно, верно также то, что во многих областях компьютеры все еще не добились значительного успеха (это еще мягко сказано), включая поставленную Тьюрингом задачу ведения разговора на свободную тему.

Возражения, основанные на принципах математики

Благодаря работам Тьюринга [1518] и Гёделя [566] широко известно, что на некоторые математические вопросы нельзя даже в принципе найти ответ в рамках конкретных формальных систем. При этом наибольшую известность получила теорема Гёделя о неполноте (см. раздел 9.5). Кратко эту теорему можно сформулировать таким образом: для любой формальной аксиоматической системы F , достаточно мощной для того, чтобы в ней можно было представить арифметику, возможно сконструировать так называемое “предложение Гёделя” $G(F)$ с описанными ниже свойствами.

- $G(F)$ — это предложение системы F , но оно не может быть доказано в рамках F .
- Если система F является непротиворечивой, то предложение $G(F)$ — истинно.

Философы, в том числе Дж. Р. Лукас [960], утверждали, что эта теорема показывает, будто машины как мыслящие субъекты всегда будут стоять ниже людей, по-

скольку машины — это формальные системы, ограниченные в силу теоремы о неполноте (они не способны установить истинность относящегося к ним самим предложения Гёделя), а люди не имеют такого ограничения. Споры вокруг данного утверждения продолжались несколько десятилетий и породили огромное количество литературы, в том числе две книги математика сэра Роджера Пенроуза [1204], [1205], повторившего это утверждение с некоторыми новыми выкрутасами (такими как гипотеза, согласно которой человек отличается от машины, поскольку его мозг действует на основе квантовой гравитации). В этой главе мы рассмотрим только три из основных проблем, связанных с этим утверждением.

Во-первых, теорема Гёделя о неполноте распространяется только на формальные системы, достаточно мощные для того, чтобы в них можно было представить арифметику. К таким формальным системам относятся машины Тьюринга, поэтому утверждение Лукаса частично основано на том предположении, что компьютеры представляют собой машины Тьюринга. Это — хорошая аппроксимация, но не совсем оправданная. Машины Тьюринга являются бесконечными, а компьютеры конечны, поэтому любой компьютер может быть описан как (очень большая) система в пропозициональной логике, на которую не распространяется теорема Гёделя о неполноте.

Во-вторых, агенту не следует стыдиться того, что он не может определить истинность некоторого высказывания, тогда как другие агенты могут. Рассмотрим приведенное ниже предложение.

Дж.Р. Лукас не может неоспоримо утверждать, что это предложение истинно.

Если бы Лукас подтвердил истинность этого предложения, то он бы противоречил самому себе, поэтому Лукас не может неоспоримо подтвердить истинность данного предложения, а это означает, что оно должно быть истинным. (Это предложение не может быть ложным, поскольку, если бы Лукас не мог неоспоримо его подтвердить, то оно было бы истинным.) Таким образом, мы продемонстрировали, что есть такое предложение, которое Лукас не может неоспоримо подтвердить, тогда как другие люди (и машины) могут. Но из-за этого никто не вправе изменить своего мнения о Лукасе в худшую сторону. В качестве еще одного примера укажем, что ни один человек за всю свою жизнь не сможет вычислить сумму 10 миллиардов десятизначных чисел, а компьютер способен выполнить такую операцию за секунды. Тем не менее мы не рассматриваем этот факт как свидетельство фундаментального ограничения способности человека мыслить. Люди вели себя интеллектуально за тысячи лет до того, как изобрели математику, поэтому маловероятно, что способность формировать математические рассуждения играет более чем периферийную роль в том, что подразумевается под понятием интеллектуальности.

В-третьих (и это — наиболее важное возражение), даже если принять предположение, что компьютеры ограничены в том, что они способны доказать, нет никаких оснований считать, будто эти ограничения не распространяются на людей. Слишком легко вести этот спор, строго доказав, что формальная система не может выполнить действие X, а затем сообщив, что люди могут выполнить действие X, используя свои человеческие неформальные методы, но не дав ни одного свидетельства в пользу этого утверждения. И действительно, невозможно доказать, что на формальные рассуждения, проводимые людьми, не распространяется теорема Гёделя о неполноте, поскольку любое строгое доказательство само должно содержать формализацию способностей человеческого гения, которые, как многие утверждают, являются не-

формализуемым, и поэтому должно опровергать само себя. Таким образом, нам остается лишь прибегать к интуитивному представлению о том, что люди иногда способны проявлять сверхчеловеческие черты математического прозрения. Подобные утверждения выражаются в виде доводов: “мы должны быть уверены в том, что мы мыслим правильно, поскольку иначе мышление вообще становится невозможным” [961]. Но если об этом зашла речь, то давно известна склонность людей совершать ошибки. Это, безусловно, касается повседневной мыслительной деятельности, но справедливо также в отношении плодов математических рассуждений, полученных в результате упорной работы. Одним из известных примеров является теорема о раскраске карты четырьмя цветами. Математик Альфред Кемпе опубликовал в 1879 году доказательство этой теоремы, которое было широко признано и стало одним из поводов к избранию этого математика в состав членов Королевского общества. Но в 1890 году Перси Хивуд указал на ошибку в этом доказательстве, и теорема оставалась недоказанной до 1977 года.

Довод, исходящий из неформализуемости

Одно из наиболее важных и трудно оспоримых критических замечаний в адрес искусственного интеллекта как сферы приложения человеческих усилий было сформулировано Тьюрингом как довод, основанный на “неформализуемости поведения”. По сути это критическое замечание сводится к утверждению, что человеческое поведение является слишком сложным для того, чтобы его можно было описать с помощью какого-либо простого набора правил, а поскольку компьютеры не способны ни на что, кроме выполнения множества правил, они не способны и проявлять такое же интеллектуальное поведение, как люди. В искусственном интеллекте неспособность выразить все, что потребуется, в виде множества логических правил называют **проблемой спецификации** (см. главу 10).

Основными сторонниками этих взглядов были философы Хьюберт Дрейфус, который написал ряд влиятельных критических статей против искусственного интеллекта, в том числе “*What Computers Can't Do*” (Что не способны делать компьютеры) [414] и “*What Computers Still Can't Do*” [415], и его брат Стюарт, совместно с которым Хьюберт написал статью “*Mind Over Machine*” [416].

Положение дел, которое критиковали эти ученые, получило известность как “добрый старый искусственный интеллект”, или сокращенно GOFAI (Good Old-Fashioned AI); этот термин был предложен Хоглендом [631]. При этом предполагается, что в основе GOFAI лежит утверждение, будто все интеллектуальное поведение может быть представлено с помощью системы, которая формирует логические рассуждения на основании множества фактов и правил, описывающих рассматриваемую проблемную область. Поэтому GOFAI соответствует простейшему логическому агенту, описанному в главе 7. Дрейфус был прав, утверждая, что логические агенты действительно имеют слабое место, поскольку не позволяют решить проблему спецификации. Но как было показано в главе 13, для использования в открытых проблемных областях в большей степени подходят вероятностные системы формирования рассуждений. Поэтому критические замечания Дрейфуса относятся не к компьютерам как к таковым, а скорее к одному конкретному способу их программирования. Однако, вполне можно предположить, что более правильное название для статьи Дрейфуса, “*What First-Order Logical Rule-Based Systems Without Learning Can't Do*” (Что не способны делать

системы на основе правил логики первого порядка, в которых не применяются средства обучения), было бы гораздо менее впечатляющим.

Согласно взглядам Дрейфуса, человеческий опыт подразумевает наличие знаний о некоторых правилах, но лишь в качестве “целостного контекста” (или “основы”), в рамках которого действуют люди. Он приводит пример корректного социального поведения при вручении и получении подарков: “Обычно люди, вручая подходящий к случаю подарок, действуют в рамках сложившихся в данном случае обстоятельств”. Очевидно, что люди обладают “непосредственным пониманием того, как следует действовать и чего следует ожидать”. Такое же утверждение он выдвигает в контексте игры в шахматы: “Шахматисту среднего уровня может потребоваться обдумать следующий ход, а гроссмейстер просто видит, что положение на доске само требует определенного хода ...правильный ответ сам складывается в его голове”. Безусловно, нельзя отрицать, что основная часть мыслительных процессов лица, готовящего подарок, или гроссмейстера, выбирающего ход, осуществляется на уровне, недоступном для самоанализа со стороны пытливого разума. Но из этого не следует, что сами эти мыслительные процессы не происходят. Важный вопрос, на который не отвечает Дрейфус, состоит в том, как правильный ход появляется в голове гроссмейстера. Напомним читателю один из комментариев Дэниела Деннета [389], приведенный ниже.

Создается впечатление, будто философы взяли на себя роль толкователей приемов фокусников. Когда их спрашивают, как фокусник ставит свой трюк с распиливанием ассистентки пополам, они объясняют, что в этом нет ничего сложного: фокусник фактически никого не распиливает; он просто заставляет людей верить, что он это делает. Если же философов спрашивают: “Но как ему удастся создать такое впечатление?”, они отвечают: “Мы в этом не компетентны”.

В статье братьев Дрейфус [416] предложен пятиэтапный процесс приобретения опыта, который начинается с обработки полученной информации на основе правил (осуществляемой по такому же принципу, как в GOFAI) и заканчивается приобретением способности мгновенно выбирать правильные ответы. Но внося свое предложение, братья Дрейфус по сути перешли из разряда критиков искусственного интеллекта в разряд его теоретиков — они предложили архитектуру нейронной сети, организованную в виде огромной “библиотеки примеров”, указав при этом на несколько проблем. К счастью, все указанные ими проблемы полностью решены, причем некоторые частично, а другие полностью. Упомянутые братьями Дрейфус проблемы перечислены ниже.

1. Качественное обобщение на основании примеров не может быть достигнуто без фоновых знаний. Дрейфусы утверждают, что не существует способа, позволяющего использовать фоновые знания в процессе обучения нейронной сети. А в действительности, как было описано в главе 19, уже разработаны такие методы, которые позволяют использовать априорные знания в алгоритмах обучения. Тем не менее эти методы основаны на том, что в наличии имеются знания, представленные в явной форме, а этот подход братья Дрейфус упорно отрицают. По мнению авторов настоящей книги, взгляды этих ученых являются основательной причиной для серьезного перепроектирования современных моделей нейронной обработки информации, для того чтобы в них можно было воспользоваться знаниями, полученными ранее в процессе обу-

чения, по такому же принципу, как такие знания используются в других алгоритмах обучения.

2. Обучение нейронной сети представляет собой одну из разновидностей контролируемого обучения (см. главу 18), для которой требуется заблаговременное выявление релевантных входных данных и правильных выходных данных. Поэтому, по утверждению братьев Дрейфус, система обучения нейронной сети не может действовать автономно, без помощи учителя-человека. В действительности обучение без учителя может осуществляться с помощью методов **неконтролируемого обучения** (см. главу 20) и **обучения с подкреплением** (см. главу 21).
3. Производительность алгоритмов обучения снижается при использовании большого количества характеристик, а если выбрано лишь подмножество характеристик, то, по словам этих ученых, “не существует способа введения новых характеристик в том случае, если будет обнаружено, что текущее множество не позволяет учитывать некоторые факты, усваиваемые в процессе обучения”. В действительности с большими множествами характеристик очень успешно справляются такие новые методы, как машины поддерживающих векторов. Как было показано в главе 19, существует также принципиальная возможность вырабатывать новые характеристики, хотя для этого требуется гораздо больше усилий.
4. Мозг способен направлять свои сенсоры на поиск релевантной информации и обрабатывать ее для извлечения аспектов, релевантных для текущей ситуации. Но Дрейфусы утверждают, что “в настоящее время неизвестны детали этого механизма и нет даже таких гипотез о его работе, которые направили бы исследования искусственного интеллекта по правильному пути”. В действительности проблеме выбора правильной ориентации сенсоров посвящена область активного зрения, основанная на теории стоимости информации (см. главу 16), а полученные теоретические результаты уже были применены при создании некоторых роботов.

В конечном итоге многие проблемы, на которых сосредоточились братья Дрейфус (фоновые обыденные знания, проблема спецификации, неопределенность, обучение, компилированные формы средств принятия решений, важность применения агентов, реагирующих на текущую ситуацию, а не бестелесных машин логического вывода), уже были решены, а достигнутые результаты воплощены в стандартных проектах интеллектуальных агентов. По мнению авторов настоящей книги, это — свидетельство прогресса искусственного интеллекта, а не подтверждение несуществимости поставленной перед ним цели.

26.2. Сильный искусственный интеллект: МОГУТ ЛИ МАШИНЫ ПО-НАСТОЯЩЕМУ МЫСЛИТЬ?

Многие философы утверждали, что даже машина, которая пройдет тест Тьюринга, все равно фактически будет не мыслить, а лишь имитировать мышление. Тьюринг предвидел и это возражение против искусственного интеллекта. В частно-

сти, он процитировал приведенный ниже фрагмент речи профессора Джеффри Джефферсона [726].

Мы сможем согласиться, что машина равна мозгу, лишь после того, как она будет в состоянии написать сонет или сочинить концерт под воздействием своих мыслей и эмоций, а не благодаря случайному совпадению нужных символов; под этим подразумевается, что машина должна не только написать подобное произведение, но и понимать, что оно ею написано.

Тьюринг назвал это возражение доводом, основанным на понятии **сознания**; согласно этому доводу, машина должна понимать свои собственные психические состояния и действия. Безусловно, сознание — это важная тема, но ключевая идея Джефферсона фактически касается проблемы **феноменологии**, или изучения непосредственного опыта, т.е. этот ученый требует, чтобы машина действительно ощущала эмоции. Другие ученые сосредоточиваются на проблеме **целенаправленности**, т.е. на вопросе о том, действительно ли приписываемые машине убеждения, желания и другие внутренние представления касаются “чего-то”, существующего в реальном мире.

Ответ Тьюринга на это возражение весьма интересен. Он мог продемонстрировать причины, по которым машины на самом деле способны были бы действовать сознательно (либо с точки зрения феноменологии, либо с точки зрения целенаправленности). Вместо этого он указал, что данный вопрос столь же некорректен, как и вопрос о том, могут ли машины мыслить. К тому же, на каком основании мы требуем применения к машинам более высоких стандартов, чем к людям? В конечном итоге в повседневной жизни мы никогда не получаем каких-либо прямых свидетельств о внутреннем психическом состоянии других людей. Тем не менее Тьюринг заявил: “Вместо ведения бесконечных споров на эту тему обычно принято заключать **джентльменское соглашение** и считать, что мыслят все”.

Тьюринг утверждает, что Джефферсон согласился бы распространить это джентльменское соглашение на машины, только если бы имел опыт общения с теми из них, которые действуют интеллектуально. Он процитировал действительно происходивший приведенный ниже диалог человека с машиной, который считается такой неотъемлемой частью передающихся из уст в уста легенд искусственного интеллекта, что мы просто обязаны его включить в эту главу.

Человек. In the first line of your sonnet which reads “shall I compare thee to a summer’s day”, would not a “spring day” do as well or better? (В первой строке вашего сонета сказано “я хочу сравнить вас с летним днем”; не было бы так же хорошо или даже лучше сказать “с весенним днем”?)

Машина. It wouldn’t scan. (Нарушилась бы ритмика.)

Человек. How about “a winter’s day”. That would scan all right. (А как насчет слов “с зимним днем”. Ритмика бы не нарушилась.)

Машина. Yes, but nobody wants to be compared to a winter’s day. (Да, но никто не хочет, чтобы его сравнивали с зимним днем.)

Человек. Would you say Mr. Pickwick reminded you of Christmas? (Вы хотите сказать, что мистер Пиквик напомнил вам о Рождестве?)

Машина. In a way. (В определенном смысле.)

Человек. Yet Christmas is a winter’s day, and I do not think Mr. Pickwick would mind the comparison. (Но все же Рождество — зимний день, и я не думаю, что мистер Пиквик возражал бы против такого сравнения.)

Машина. I don't think you're serious. By a winter's day one means a typical winter's day, rather than a special one like Christmas. (Я не думаю, что вы говорите серьезно. Под зимним днем подразумевается обычный зимний день, а не такой особый день, как Рождество.)

В заключение Тьюринг отметил, что вопрос о сознании является трудноразрешимым, но опроверг мнение о том, что он имеет большую значимость для практики искусственного интеллекта: “Я отнюдь не желаю, чтобы мои слова были истолкованы таким образом, будто я не считаю проблему сознания сложной загадкой... но я не думаю, что нужно обязательно разгадать все подобные загадки, прежде чем мы сможем ответить на тот вопрос, о котором идет речь в данной статье”. Авторы настоящей книги согласны с Тьюрингом: мы заинтересованы в создании программ, которые действуют интеллектуально, а не в том, чтобы дать кому-то повод считать эти действия настоящими или имитированными. С другой стороны, эта проблема остается предметом острого интереса для многих философов. Для того чтобы понять суть такой заинтересованности, рассмотрим вопрос о том, считаются ли реальными другие искусственно созданные объекты.

В 1848 году Фредерик Вёлер впервые синтезировал искусственную мочевину. Это достижение было очень важным, поскольку стало доказательством единства органической и неорганической химии, а также позволило поставить точку в вопросе, который до сих пор был предметом горячих дебатов. После успешного осуществления этого синтеза химики согласились, что искусственная мочевина действительно представляет собой мочевину, поскольку обладает всеми правильными физическими свойствами. Аналогичным образом, нельзя отрицать, что искусственные подслащивающие вещества действительно являются подслащивающими веществами, а искусственное оплодотворение (еще один термин с аббревиатурой AI — Artificial Insemination) действительно является оплодотворением. С другой стороны, искусственные цветы — это не цветы, и, как указал Дэниел Деннет (Daniel Dennett), искусственное вино Шато Латур — это не вино Шато Латур, даже если образцы того и другого нельзя отличить друг от друга с помощью химического анализа, просто потому, что оно не было изготовлено в должном месте правильным способом. А искусственно выполненный рисунок Пикассо — это не рисунок Пикассо, независимо от того, похож он на оригинал или нет.

На основании изложенного можно сделать вывод, что в некоторых случаях важно лишь поведение искусственного объекта, а в других случаях играет роль также происхождение искусственного объекта. То, в каком случае приобретает важность последний фактор, по-видимому, обусловлено лишь принятыми соглашениями. А когда речь идет об искусственном разуме, мы не можем опереться на принятое по этому поводу соглашение, и нам остается полагаться лишь на интуитивные предположения. Философ Джон Сирл [1376, с. 37, 38] выдвинул следующее весьма убедительное предположение.

Никто не думает, что компьютерная имитация грозы заставит его вымокнуть... так почему же люди, будучи в здравом уме, могут предположить, что компьютерная имитация мыслительных процессов действительно представляет собой мыслительные процессы?

Безусловно, нельзя не согласиться, что компьютерные имитации гроз не заставят нас вымокнуть, но не совсем понятно, как можно перенести эту аналогию на компьютерные имитации мыслительных процессов. К тому же создаваемые в Голливуде имитации гроз, в которых используются опрыскиватели и воздушные подушки, действительно заставляют актеров вымокнуть. Большинство людей, не задумываясь, скажет,

что компьютерная имитация сложения является сложением, а компьютерная имитация шахматной игры является шахматной игрой. Что больше напоминают мыслительные процессы — грозы или абстрактные операции, такие как арифметическое сложение и игра в шахматы? С чем их следует сравнивать — со штучными изделиями, такими как вино Шато Латур и картины Пикассо, или с массовой продукцией, такой как мочевины? Ответы на все эти вопросы зависят от принятой теории психических состояний и процессов.

В теории **функционализма** утверждается, что психическим состоянием является любое промежуточное причинное условие, связывающее входные и выходные данные. Согласно теории функционализма, любые две системы с изоморфными причинными процессами должны иметь одни и те же психические состояния. Поэтому компьютерная программа может иметь такие же психические состояния, как и человек. Безусловно, мы еще не дали определения того, что действительно подразумевается под термином “изоморфный”, но основное допущение состоит в том, что существует некоторый уровень абстракции, ниже которого конкретные детали реализации не имеют значения; при условии, что ниже этого уровня процессы являются изоморфными, возникают одни и те же психические состояния.

В отличие от этого, в теории **биологического натурализма** утверждается, что психические состояния представляют собой высокоуровневые эмерджентные характеристики, которые вызваны неврологическими процессами низкого уровня в нейронах, и ведущую роль в этих процессах играют некоторые (неопределенные) свойства нейронов. Это означает, что психические состояния не могут быть продублированы лишь на основе некоторой программы, имеющей такую же функциональную структуру и проявляющей такое же поведение, выраженное в виде входных/выходных данных; мы должны потребовать, чтобы эта программа эксплуатировалась в архитектуре, обладающей такой же причинной мощностью, как и нейроны. В указанной теории ничего не говорится о том, почему нейроны обладают этой причинной мощностью, а также о том, существуют ли другие физические воплощения, которые могут иметь или не иметь эту причинную мощь.

Чтобы проанализировать эти две точки зрения, вначале рассмотрим одну из самых старых проблем в области философии разума, а затем обратимся к трем мысленным экспериментам.

Проблема разума и тела

Проблема разума и тела касается вопроса о том, как психические состояния и процессы связаны с физическими состояниями и процессами (а именно с процессами, происходящими в мозгу). Игнорируя достаточно высокую сложность этой проблемы, обобщим ее до уровня проблемы “архитектуры разума”, что позволит нам вести речь о том, могут ли машины иметь разум.

Почему проблема разума и тела является такой сложной? Первую сложность обнаружил еще Рене Декарт, который размышлял над тем, как бессмертная душа взаимодействует со смертным телом, и пришел к выводу, что душа и тело — это два различных типа субстанций; в этом состоит так называемая теория **дуализма**. С другой стороны, теория **монизма**, часто называемая **материализмом**, основана на том, что таких субстанций, как нематериальные души, просто не бывает; в мире имеются только материальные объекты. Поэтому все психические состояния

(возникающие, когда человек испытывает боль, осознает себя как скачущий на лошади или думает, что Вена — столица Австрии) представляют собой состояния мозга. Джон Сирл метко сформулировал эту идею в виде лозунга “Мозг рождает разум”.

Но материализму приходится сталкиваться с двумя серьезными препятствиями. Первым из них является проблема **свободной воли**: как так может оказаться, что чисто физический разум, каждое преобразование в котором строго управляется законами физики, все еще сохраняет какую-то свободу выбора? Большинство философов рассматривают эту проблему как требующую тщательного переопределения наших наивных представлений о свободной воле, а не представляющую собой какое-либо покушение на материализм. Вторым препятствием является проблема, касающаяся общего вопроса о **сознании** (а также связанных с ним, но не идентичных вопросам **понимания** и **самосознания**). В наиболее простой формулировке этот вопрос состоит в том, почему человек ощущает себя как имеющий определенные мыслительные состояния и вместе с тем не чувствует себя как имеющий какие-то другие физические состояния (например, не считает себя камнем).

Чтобы приступить к поиску ответа на подобные вопросы, мы должны найти способы вести речь о состояниях мозга на уровнях, более абстрактных, чем конкретные конфигурации всех атомов мозга определенного человека в конкретное время. Например, когда я размышляю о столице Австрии, мой мозг подвергается бесчисленному количеству мельчайших изменений за время, прошедшее от одной пикосекунды до другой, но это не приводит к качественному изменению состояния мозга. Для того чтобы учесть возможные ситуации, необходимо ввести понятие *типов состояния мозга*, в рамках которого можно было бы судить, принадлежат ли два состояния мозга к одному и тому же или к разным типам. Различные ученые имеют несовпадающие взгляды на то, что подразумевается в данном случае под типом. Но почти все ученые считают, что если взять живой мозг и заменить в нем некоторые из атомов углерода новым множеством атомов углерода², то психическое состояние мозга не изменится. Эта гипотеза вполне оправдана, поскольку в действительности в мозгу непрерывно происходит замена атомов в результате метаболических процессов, тем не менее такой процесс, по-видимому, не вызывает существенных психических расстройств.

Теперь рассмотрим конкретный вид психического состояния: **пропозициональные позиции** (впервые описанные в главе 10), которые также известны как **ментальные состояния**. Таковыми являются состояния, подобные убежденности, уверенности, желанию, чувству страха и т.д., которые относятся к некоторому аспекту внешнего мира. Например, уверенность в том, что Вена — столица Австрии, — это убеждение, касающееся конкретного города и его статуса. Нас интересует вопрос, могут ли компьютеры иметь ментальные состояния, чтобы можно было понять, как охарактеризовать эти состояния. Например, можно утверждать, что психическое состояние, в котором я хочу гамбургер, отличается от состояния, в котором я хочу пиццу, поскольку гамбургер и пицца в реальном мире отличаются друг от друга. А если верно такое утверждение, то ментальные состояния имеют необходимую связь с относящимися к ним объектам во внешнем мире. С другой стороны, всего лишь за несколько абзацев перед этим было сформулировано утверждение, что психические состояния представляют собой состояния мозга, поэтому идентичность или неиден-

² Возможно, даже атомами другого изотопа углерода, как иногда бывает в экспериментах по сканированию мозга.

тичность психических состояний должна определяться полностью “внутри самой головы”, без ссылок на реальный мир. Чтобы лучше изучить эту дилемму, обратимся к мысленному эксперименту, позволяющему отделить целенаправленные состояния от относящихся к ним внешних объектов.

Эксперимент “мозг в колбе”

Допустим, что при желании мозг человека можно отделить от тела сразу после рождения и поместить в колбу, искусно спроектированную для этой цели. В этой колбе мозг получает питание и опору, а она позволяет ему расти и развиваться. Наряду с тем в мозг подаются электронные сигналы от компьютера, моделирующего полностью вымышленный мир, а моторные сигналы от мозга перехватываются и используются для модификации этой имитации должным образом³. В таком случае мозг может иметь психическое состояние *DyingFor (Me, Hamburger)* (страстное желание получить гамбургер), даже несмотря на то, что у него нет тела, которое испытывало бы чувство голода, а также вкусовых рецепторов, чтобы ощутить вкус, к тому же в реальном для мозга (но фактически имитируемом) мире могло бы просто не оказаться гамбургера. Было бы это психическое состояние таким же, какое испытывает мозг в живом теле?

Один из способов разрешения этой дилеммы состоит в использовании гипотезы о том, что содержимое психических состояний можно интерпретировать с двух разных точек зрения. В подходе на основе **широкого анализа содержимого** психические состояния интерпретируются с точки зрения всезнающего внешнего наблюдателя, имеющего доступ к информации обо всей ситуации, который способен замечать любые различия в мире. Поэтому при широком анализе содержимого чувства мозга, живущего в колбе, можно отличить от чувств “обычного” человека. А при **узком анализе содержимого** учитывается только внутренняя субъективная точка зрения, и при таком подходе чувства, испытываемые тем и другим мозгом, остаются одинаковыми.

Уверенность в том, что гамбургер — желанная пища, имеет определенный характер связи с конкретным объектом, поскольку обязательно должен быть кто-то, обладающий уверенностью в таком свойстве гамбургера. Переходя к рассуждениям такого рода, мы вступаем в область познания **качества**, или собственного опыта (в англоязычной литературе для обозначения этого понятия применяется термин “qualia”, происходящий от латинского слова, которое переводится примерно как “именно такие”). Предположим, что в результате какого-то нарушения в нервных путях сетчатки и мозга неким лицом X воспринимается как красный тот цвет, который лицо Y воспринимает как зеленый, и наоборот. В таком случае, увидев один и тот же сигнал светофора, оба они действуют одинаково, но воспринимаемый ими опыт должен быть в определенной степени разным. Оба эти лица могут согласиться, что полученные ими данные восприятия говорят о том, что “свет на светофоре — красный”, но сами эти восприятия остаются разными. Поэтому неясно, свидетельствует ли это о том, что они имеют одинаковые или разные психические состояния.

³ Описанная ситуация может быть знакома тем, кто смотрел фильм *The Matrix* (Матрица), вышедший на экраны в 1999 году.

Теперь перейдем еще к одному мысленному эксперименту, который относится к вопросу о том, могут ли иметь психические состояния физические объекты, отличные от нейронов человека.

Эксперимент с протезом мозга

Мысленный эксперимент с протезом мозга был предложен в середине 1970-х Кларком Глаймором и описан в трудах Джона Сирла [1376], но его чаще всего связывают с работой Ханса Моравека [1084]. Этот эксперимент заключается в следующем: предположим, что нейрофизиология достигла огромного уровня развития, на котором существует идеальное понимание взаимодействия входных и выходных сигналов и связей между всеми нейронами в мозгу человека. Предположим также, что существует возможность создавать микроскопические электронные устройства, имитирующие поведение нейрона, которые можно незаметно для человека подключать к его нервной ткани. Наконец, предположим, что некая чудесная хирургическая техника позволяет заменять отдельные нейроны соответствующими электронными устройствами, не нарушая работу мозга в целом. Эксперимент состоит в постепенной замене всех нейронов в голове человека электронными устройствами, а затем в обратном выполнении этого процесса для возврата испытуемого субъекта в его нормальное биологическое состояние.

Нас интересует как внешнее поведение, так и внутренний опыт этого субъекта во время операции и после нее. Согласно определению этого эксперимента, внешнее поведение субъекта должно оставаться неизменным по сравнению с тем, которое наблюдалось бы в том случае, если бы эта операция не выполнялась⁴. Итак, несмотря на то, что в присутствии или отсутствии сознания не так уж легко убедиться, будучи сторонним наблюдателем, сам субъект эксперимента должен по крайней мере иметь способность зарегистрировать любые изменения в своем восприятии собственного сознания. Очевидно, что возникает прямой конфликт интуитивных представлений о том, что может произойти. Моравек, будучи специалистом в области робототехники и приверженцем взглядов функционалистов, убежден в том, что сознание субъекта, подвергающегося эксперименту, останется незатронутым, а Сирл, философ и натуралист-биолог, столь же твердо убежден, что сознание у субъекта эксперимента постепенно исчезнет, о чем свидетельствует приведенная ниже цитата из его работы.

Вы обнаружите, к своему полному изумлению, что действительно теряете контроль над своим внешним поведением. Например, вы заметите, что при проверке врачами вашего зрения вам, допустим, скажут: “Мы держим перед вами объект красного цвета; пожалуйста, сообщите нам, что вы видите”. Вы захотите крикнуть: “Я ничего не вижу. Я полностью ослеп”, но услышите, как ваш голос говорит, полностью не подчиняясь вашему контролю: “Я вижу перед собой объект красного цвета...” Ваше сознание постепенно сужается и исчезает, тогда как внешне наблюдаемое поведение остается неизменным [1379].

Но существует возможность вести этот спор, опираясь не только на интуицию. Во-первых, следует отметить, что внешнее поведение будет оставаться одинаковым в процессе того, как субъект постепенно теряет сознание, только в том случае, если

⁴ Можно также представить себе, что в эксперименте участвует идентичный “контрольный” субъект, над которым якобы выполняется операция, но фактически она не проводится. Это позволяет сравнивать поведение двух субъектов.

воля субъекта исчезает мгновенно и полностью; в противном случае сужение сознания должно было бы отразиться на внешнем поведении; это означает, что испытуемый должен был бы закричать: “Помогите, я теряю сознание!” или нечто в этом роде. Такая гипотеза мгновенного исчезновения воли в результате постепенной замены одного за другим отдельных нейронов кажется маловероятной.

Во-вторых, рассмотрим, что произойдет, если мы будем задавать субъекту вопросы, касающиеся того, как он сам ощущает наличие у него сознания в тот период, когда у него не останется ни одного настоящего нейрона. Согласно условиям эксперимента, мы должны получать примерно такие ответы: “Я чувствую себя прекрасно. Я должен также отметить, что немного удивлен, поскольку верил в истинность доводов Сирла”. Еще один вариант состоит в том, что мы могли бы уколоть субъекта заостренной палочкой и услышать ответ: “Ой, больно”. Теперь, если вернуться к обычной жизни, то скептик может возразить, что подобные выходные данные могут быть получены и от программ искусственного интеллекта как простые результаты принятых соглашений. Действительно, совсем не сложно предусмотреть, например, такое правило: “Если на датчике номер 12 появится сигнал с высокой интенсивностью, то выдать выходные данные «Ой, больно»”, но весь смысл рассматриваемого эксперимента состоит в том, что мы продублировали функциональные свойства обычного человеческого мозга, поэтому предполагается, что электронный мозг не содержит таких структур, в которых реализованы подобные соглашения. Это означает, что нужно как-то объяснить, чем обусловлены проявления сознания, вырабатываемые электронным мозгом, которые основаны лишь на функциональных свойствах нейронов. И это объяснение должно также распространяться на настоящий мозг, который имеет такие же функциональные свойства. На наш взгляд, можно сделать только два приведенных ниже возможных вывода.

1. Причинные механизмы формирования сознания, которые вырабатывают выходные данные такого рода в обычном мозгу, все еще продолжают действовать в электронной версии мозга, поэтому последняя также обладает сознанием.
2. Осознаваемые психические события в обычном мозгу не имеют причинной связи с поведением, а поскольку они отсутствуют также в электронном мозгу, последний не обладает сознанием.

Хотя нельзя исключить вторую возможность, при данном подходе сознание сводится к тому, что философы называют *эпифеноменальной* (выраженной в качестве побочного явления) ролью — как будто что-то происходит, но не отбрасывает тени, которая должна была бы появиться в наблюдаемом мире. Кроме того, если сознание действительно эпифеноменально, то в мозгу должен находиться второй, бессознательный механизм, с помощью которого и формируется восклицание “Ой, больно”, когда человека колют заостренной палочкой.

В-третьих, рассмотрим ситуацию, возникшую после того, как операция проделана в обратном направлении и субъект снова имеет обычный мозг. И в этом случае внешнее поведение субъекта должно быть, по определению, таким же, как если бы операция вовсе не проводилась. В частности, мы были бы вправе задать субъекту такие вопросы: “Что вы чувствовали во время операции? Помните, как вас укололи заостренной палочкой?” Субъект должен точно помнить фактический характер своего осознанного опыта, включая качественную сторону этого опыта, несмотря на тот факт, что, согласно Сирлу, такого опыта не должно быть.

Сирл мог бы возразить, что мы не определили эксперимент должным образом. Если бы настоящие нейроны, скажем, прекращали действовать в том промежутке времени, когда они были извлечены, а затем снова помещены в мозг, то, безусловно, они не могли бы “запомнить” опыт, полученный в это время. Чтобы учесть это обстоятельство, необходимо обеспечить обновление состояния нейронов в соответствии с изменением внутреннего состояния искусственных нейронов, которыми они заменялись. Если бы в таком случае предполагалось наличие “нефункциональных” аспектов реальных нейронов, которые привели бы к поведению, функционально отличному от того, которое наблюдалось, пока искусственные нейроны были бы еще на месте, то налицо было бы простое приведение к абсурду, поскольку означало бы, что искусственные нейроны функционально не эквивалентны настоящим нейронам (одно из возможных возражений против этого довода приведено в упр. 26.3).

Патрисия Чарчленд [260] указала, что приведенные выше доводы, основанные на теории функционализма и применяемые на уровне нейронов, могут также использоваться на уровне любой более крупной функциональной единицы — группы нейронов, раздела мозга, доли, полушария или целого мозга. Это означает, что, согласившись с доводом, что эксперимент с протезом мозга демонстрирует наличие сознания у мозга, в котором нейроны заменены электронными компонентами, мы должны также согласиться, что сознание сохраняется, если весь мозг заменяется схемой, в которой входные данные отображаются на выходные с помощью огромной поисковой таблицы. Такое представление неприемлемо для многих людей (включая самого Тьюринга), интуиция которых подсказывает, что поисковые таблицы вряд ли могут иметь сознание или, по меньшей мере, что сознательный опыт, сформированный во время поиска в таблице, не сопоставим с опытом, сформированным в процессе работы системы, которая может быть описана (даже в примитивном, вычислительном смысле) как манипулирование с убеждениями, результатами самоанализа, целями и тому подобными явлениями, которые формируются мозгом. Эти замечания свидетельствуют о том, что эксперимент с протезом мозга может быть эффективным средством, подкрепляющим нашу интуицию, только если в нем не предусматривается замена сразу всего мозга, но это и не означает, что в данном эксперименте может лишь рассматриваться замена одних атомов другими, как хочет нас заставить считать Сирл.

Китайская комната

Последний мысленный эксперимент, который будет описан в данной главе, по-видимому, является самым известным из всех. Идея этого эксперимента принадлежит Джону Сирлу [1376], описавшему гипотетическую систему, в отношении которой любому наблюдателю ясно, что она работает под управлением какой-то программы и успешно проходит тест Тьюринга, но также ясно (согласно Сирлу), что эта система не понимает смысла каких-либо из ее входных или выходных данных. На основании этого Сирл делает вывод, что работа системы под управлением приемлемой программы (т.е. программы, вырабатывающей правильные выходные данные) не является достаточным условием для обладания разумом.

Система состоит из человека, который понимает только английский язык, снабжен книгой с правилами, написанной на английском языке, а в его распоряжении находятся разные стопки бумаг, причем некоторые из них пусты, а другие заполне-

ны описаниями, не поддающимися расшифровке. (Таким образом, человек играет роль процессора компьютера, книга правил представляет собой программу, а стопки бумаг соответствуют запоминающему устройству.) Система находится в комнате с небольшим отверстием, выходящим наружу. Через отверстие появляются полоски бумаги с символами, не поддающимися расшифровке. Человек находит совпадающие с ними символы в книге правил и следует обнаруженным в ней инструкциям. Инструкции могут предусматривать задания по написанию символов на новых полосках бумаги, поиску символов в стопках, переупорядочению стопок и т.д. В конечном итоге инструкции диктуют необходимость написания одного или нескольких символов на листе бумаги, который снова передается во внешний мир.

До сих пор все шло нормально. Но из внешнего мира мы видим систему, которая принимает входные данные в форме предложений на китайском языке и формирует ответы на китайском, которые внешне кажутся “интеллектуальными”, как и беседа, мысленно представленная Тьюрингом⁵. После этого Сирл проводит следующие рассуждения: человек, сидящий в комнате, не понимает китайского (это дано). Книга правил и стопки бумаги, будучи просто листами бумаги, не понимают китайского. Поэтому речь не может идти о каком-либо понимании китайского языка. *Поэтому, согласно Сирлу, эксплуатация даже подходящей программы не обязательно приводит к развитию понимания.*

Как и Тьюринг, Сирл рассмотрел и попытался опровергнуть целый ряд ответов на его доводы. Некоторые комментаторы, включая Джона Маккарти и Роберта Виленского, выдвинули предложения, которые Сирл назвал *системными ответами*. Их возражение состоит в том, что человека, сидящего в комнате, безусловно, можно спросить, понимает ли он китайский язык, но это аналогично тому, как если бы процессор компьютера спросили, может ли он извлекать кубические корни. В обоих случаях ответ является отрицательным и в обоих случаях, согласно системному ответу, вся система обладает способностью, которая была предметом вопроса. Безусловно, если задать системе с китайской комнатой вопрос на китайском языке, знает ли она китайский, ответ будет утвердительным (и сформулированным на живом китайском языке). Согласно корректному соглашению Тьюринга, этого должно быть достаточно. Ответ Сирла — это возврат к той идее, что понимание не заключено в мозгу человека, сидящего в китайской комнате, и не может быть заключено в стопках бумаги, поэтому не может быть никакого понимания. Далее Сирл указывает, что можно представить себе ситуацию, когда человек запоминает книгу правил и содержимое всех стопок бумаги, поэтому больше нет ни одного объекта, которому можно было бы приписать понимание, кроме самого человека; но и после этого, если ему будет задан вопрос (на английском языке), понимает ли он китайский, ответ будет отрицательным.

Теперь перейдем к реальному анализу этой проблемы. Переход от эксперимента с использованием стопок бумаги к эксперименту с запоминанием — это просто попытка сбить читателя с толку, поскольку обе формы представляют собой варианты физического воплощения работающей программы. Фактические утверждения, выдвинутые Сирлом, опираются на четыре приведенных ниже аксиомы [1378].

⁵ Тот факт, что стопки бумаги могут оказаться больше всей нашей планеты, а выработка ответов может занять миллионы лет, не имеет отношения к логической структуре данного эксперимента. Одной из целей обучения философии является выработка тщательно отточенного понимания того, какие возражения являются обоснованными и какие нет.

1. Компьютерные программы представляют собой формальные, синтаксические сущности.
2. Разум имеет мыслительное содержание, или семантику.
3. Синтаксиса как такового не достаточно для семантики.
4. Мозг порождает разум.

На основании первых трех аксиом Сирл делает вывод, что программы не могут служить достаточным условием для появления разума. Иными словами, агент, в котором функционирует какая-то программа, может оказаться разумным, но он не обязательно становится разумным лишь в силу того, что в нем работает программа. На основании четвертой аксиомы Сирл делает вывод: “Любая другая система, способная порождать разум, должна обладать причинной мощью (по меньшей мере), эквивалентной той, какой обладает мозг”. На основании этого он делает вывод, что любой искусственный мозг должен воплощать в себе такую же причинную мощь, как и мозг, а не только работать под управлением конкретной программы, и что мозг человека не вырабатывает мыслительные феномены исключительно благодаря тому, что в нем функционирует какая-то программа.

Вывод о том, что применения программ недостаточно для создания разума, действительно следует из этих аксиом, если допускается их вольная интерпретация. Но само обоснование вывода проведено неудовлетворительно — все, что смог доказать Сирл, состоит в том, что если явно отвергнуты принципы функционализма (как было сделано в его третьей аксиоме), то заключение, согласно которому объекты, отличные от мозга, порождают разум, становится необязательным. Это предположение вполне обосновано, поэтому вся дискуссия сводится к тому, может ли быть принята третья аксиома. Согласно Сирлу, весь смысл эксперимента с китайской комнатой состоит в предоставлении интуитивного обоснования для третьей аксиомы. Но реакция других исследователей показывает, что такие интуитивные представления близки по духу только тем, кто уже был склонен соглашаться с идеей, что программы, взятые в чистом виде, не способны вырабатывать истинное понимание.

Еще раз отметим, что цель эксперимента с китайской комнатой состоит в опровержении понятия сильного искусственного интеллекта — утверждения, что эксплуатация программы подходящего типа обязательно приводит к появлению разума. Этот мысленный эксперимент проводится путем демонстрации внешне интеллектуальной системы, в которой функционирует программа подходящего типа, но в отношении этой системы, согласно Сирлу, можно явно показать, что она не обладает разумом. Для этого Сирл прибегает к интуиции, а не к доказательству; он как будто говорит нам: “достаточно взглянуть на эту комнату; разве в ней может быть разум?” Но точно такой же довод можно привести и применительно к мозгу — достаточно взглянуть на этот конгломерат клеток (или атомов), работающих вслепую в соответствии с законами биохимии (или физики); разве в нем может быть разум? Почему в куске мозга может быть разум, а в куске печени — нет?

Более того, Сирл, соглашаясь с тем, что материалы, отличные от нейронов, могут в принципе быть носителем разума, ослабляет свои доводы еще больше, по двум причинам: во-первых, нам приходится руководствоваться только интуицией Сирла (или своей собственной), чтобы доказать, что в китайской комнате отсутствует разум, и, во-вторых, даже если мы решим, что в этой комнате нет разума, такой вывод

не позволит нам узнать что-либо о том, не будет ли программа, работающая в какой-то другой физической среде (включая компьютер), иметь разум.

Сирл допускает логическую возможность того, что мозг действительно реализует программу искусственного интеллекта традиционного типа, но та же программа, работающая в машине неподходящего типа, не создает разум. Сирл отрицает то, будто он верит, что “машины не могут иметь разума”, скорее он утверждает, что некоторые машины имеют разум, например люди — это биологические машины, обладающие разумом. Но он также оставляет нас в неведении относительно того, какого же типа машины подходят или не подходят под определение понятия разумных машин.

26.3. ЭТИЧЕСКИЕ И МОРАЛЬНЫЕ ПОСЛЕДСТВИЯ РАЗРАБОТКИ ИСКУССТВЕННОГО ИНТЕЛЛЕКТА

До сих пор в этой главе в основном рассматривался вопрос о том, *может ли* быть разработан искусственный интеллект, но необходимо также посвятить время анализу вопроса, *должен ли* он все же быть разработан. Если последствия создания технологии искусственного интеллекта с большей вероятностью будут отрицательными, чем положительными, то люди, работающие в этой области, несут моральную ответственность, обязывающую их направить свои поиски в другие области. Непредвиденные отрицательные побочные эффекты стали следствием внедрения многих новых технологий: двигатели внутреннего сгорания послужили причиной загрязнения воздуха и повсеместного строительства дорог, даже в самых райских уголках; ядерные технологии стали причиной взрывов в Чернобыле и на острове Тримайл Айленд и создали угрозу глобального разрушения. Все ученые и инженеры сталкиваются с этическими соображениями, которые они должны учитывать, выполняя свою работу, выбирая проекты, которые должны или не должны быть реализованы, а также способы осуществления этих проектов. На эту тему была даже написана книга *Ethics of Computing* [103]. Но искусственный интеллект, по-видимому, становится источником некоторых невиданных ранее проблем, в том числе перечисленных ниже, кроме, скажем, строительства мостов, которые падают под собственным весом.

- В результате автоматизации может увеличиться количество безработных.
- Может уменьшиться (или увеличиться) количество свободного времени, имеющегося в распоряжении людей.
- Люди могут потерять чувство собственной уникальности.
- Люди могут потерять некоторые из своих прав на личную жизнь.
- Использование систем искусственного интеллекта может привести к тому, что люди станут более безответственными.
- Успех искусственного интеллекта может стать началом конца человеческой расы.

Рассмотрим каждую из этих проблем по очереди.

- В результате автоматизации может увеличиться количество безработных. Современная индустриальная экономика стала полностью зависимой от применения компьютеров в целом и отдельных программ искусственного интеллекта в частности. Например, значительная часть экономики, особенно в

Соединенные Штатах, зависит от доступности потребительского кредита. Проверка заявок на выпуск кредитных карт, выдача разрешений на осуществление платежей и раскрытие мошеннических сделок, а также другие операции теперь выполняются программами искусственного интеллекта. Напрашивается вывод, будто из-за появления этих программ свои места потеряли тысячи служащих, но в действительности этих рабочих мест без применения программ искусственного интеллекта просто не было бы, поскольку в случае применения ручного труда стоимость этих операций была бы неприемлемой. До сих пор автоматизация с помощью технологии искусственного интеллекта неизменно создавала больше рабочих мест, чем устраняла, к тому же приводила к появлению более интересных и высокооплачиваемых специальностей. Теперь, после того как канонической программой искусственного интеллекта стал “интеллектуальный агент”, предназначенный для помощи человеку, потеря работы становится еще менее вероятным последствием внедрения искусственного интеллекта по сравнению с той эпохой, когда все усилия специалистов по искусственному интеллекту сосредоточивались на создании “экспертных систем”, предназначенных для замены людей.

- Может уменьшиться (или увеличиться) количество свободного времени, имеющегося в распоряжении людей. Алвин Тоффлер в своей книге *Future Shock* [1510] указал: “Рабочая неделя с начала столетия сократилась на 50%. И никого не удивляет прогноз, что к 2000 году она будет сокращена еще наполовину”. Артур К. Кларк [265] писал, что “люди в 2001 году могут столкнуться с будущим, заполненным необычайной скукой, когда главной проблемой в жизни станет принятие решения о том, какой из нескольких сотен телевизионных каналов выбрать для просмотра”. Единственным из этих прогнозов, который сбывся хоть в какой-то степени, стало количество телевизионных каналов [1453]. А вместо сокращения рабочего дня люди, занятые в отраслях промышленности, характеризующихся интенсивным использованием знаний, стали чувствовать себя частью интегрированной компьютеризированной системы, которая работает 24 часа в сутки; чтобы успешно справляться со своими обязанностями, они вынуждены работать все больше и больше. В индустриальной экономике вознаграждения приблизительно пропорциональны затратами времени; увеличение продолжительности работы на 10% обычно приводит в среднем к увеличению доходов на 10%. А в информационной экономике, характеризующейся наличием широкополосной связи и упрощением тиражирования интеллектуальной собственности (которую Фрэнк и Кук [495] назвали “обществом, в котором победитель получает все”), самое большое вознаграждение приносит способность оказаться немного более успешным, чем конкурент; увеличение продолжительности работы на 10% может означать увеличение дохода на 100%. Поэтому каждый испытывает все возрастающий прессинг, который заставляет его работать все интенсивнее. Искусственный интеллект способствует увеличению темпов внедрения технологических инноваций и поэтому вносит свой вклад в эту общую тенденцию, но искусственный интеллект обещает также предоставить нам возможность снять с себя часть нагрузки и позволить нашим автоматизированным агентам хоть какое-то время выполнять работу за нас.

- Люди могут потерять чувство собственной уникальности. В своей книге *Computer Power and Human Reason* [1566] Вейценбаум, автор программы Eliza, указал на некоторые потенциальные угрозы, с которыми сталкивается общество в связи с развитием искусственного интеллекта. Одним из самых важных доводов Вейценбаума является то, что в результате исследований в области искусственного интеллекта кажется уже не такой невероятной идея о том, что люди представляют собой автоматы, а эта идея приводит к потере самостоятельности или даже человечности. Но авторы настоящей книги хотят отметить, что эта идея существовала задолго до появления искусственного интеллекта и восходит по меньшей мере к тем временам, когда вышла книга *L'Homme Machine* [875]. Авторы также отмечают, что люди пережили и другие покушения на чувство их уникальности: Коперник, автор книги *De Revolutionibus Orbium Coelestium* [294], убрал Землю из центра солнечной системы, а Дарвин, автор книги *Descent of Man* [326], поместил вид *Homo sapiens* на тот же уровень, где находятся все другие виды живых существ. Поэтому даже в случае его широкого и успешного наступления искусственный интеллект станет не большей угрозой для моральных устоев общества XXI века, чем была дарвиновская теория эволюции для XIX века.
- Люди могут потерять некоторые из своих прав на личную жизнь. Вейценбаум также указал, что развитие технологии распознавания речи может привести к широкому распространению средств прослушивания телефонных разговоров и поэтому к потере гражданских свобод. Он не предвидел, что когда-то в мире террористические угрозы станут настолько реальными, что изменят отношение людей к тому, с каким уровнем надзора они будут готовы согласиться, однако правильно понял, что искусственный интеллект обладает потенциалом к созданию средств надзора массового применения. Предсказание Вейценбаума вскоре может осуществиться⁶: правительство США рассматривает вопрос о внедрении системы Echelon, которая “состоит из сети пунктов прослушивания, антенных полей и радарных станций; система опирается на поддержку компьютеров, в которых используется языковой перевод, распознавание речи и поиск по ключевым словам для автоматического просеивания трафика, идущего по телефону, электронной почте, факсу и телексу”. Многие согласны с тем, что компьютеризация приводит к ущемлению прав на личную жизнь, — один из старших руководителей компании Sun Microsystems Скотт Макнили даже заявил: “У вас все равно нет никакой личной жизни. Забудьте о ней”. Другие с этим не согласны; например, судья Луис Брандейс писал еще в 1890 году: “Право на личную жизнь является самым всеобъемлющим из всех прав... ведь это — право оставаться самим собой”.
- Использование систем искусственного интеллекта может привести к тому, что люди станут более безответственными. В той атмосфере постоянной готовности к судебным разбирательствам, которая доминирует в Соединенных Штатах, большую важность приобретают вопросы правовой ответственности. Если терапевт принял на веру суждение медицинской экспертной системы в от-

⁶ См. статью “Eavesdropping on Europe” (Применение средств подслушивания в Европе), *Wired news*, 9/30/1998, и процитированные в ней отчеты Европейского экономического сообщества.

ношении диагноза, то кто будет нести ответственность, если диагноз окажется неправильным? К счастью, теперь общепризнано, что нельзя рассматривать как пренебрежение служебными обязанностями выполнение терапевтом медицинских процедур, которые имеют высокую ожидаемую полезность, даже если фактические результаты оказались катастрофическими для пациента (отчасти такая смена взглядов обусловлена ростом влияния методов теории решений в медицине). Поэтому приведенный выше вопрос должен рассматриваться в такой формулировке: “Кто должен нести ответственность, если диагноз оказался неоправданным?” До сих пор суды исходили из того, что медицинские экспертные системы играют такую же роль, как медицинские учебники и справочники; терапевты обязаны понять ход рассуждений, лежащий в основе любого решения системы, и использовать свое собственное суждение для принятия решения о том, нужно ли следовать рекомендациям системы. Поэтому при проектировании медицинских экспертных систем в виде интеллектуальных агентов действия этих систем следует рассматривать не как непосредственно воздействующие на пациента, а как влияющие на поведение терапевта. А если экспертные системы когда-то будут надежно вырабатывать более точные диагнозы по сравнению с людьми, врачи могут стать юридически ответственными, если они не используют рекомендации экспертной системы. Такая предпосылка рассматривается в [528].

Приобретают также важное значение аналогичные вопросы, касающиеся использования интеллектуальных агентов в Internet. Например, достигнут определенный прогресс в части внедрения ограничений в интеллектуальных агенты, чтобы они не могли, допустим, повредить файлы других пользователей [1571]. Проблемы становятся еще более важными, когда речь идет о денежных сделках. Если финансовые операции выполняются интеллектуальным агентом “от чьего-то имени”, то кто будет отвечать за причиненные убытки? Будет ли существовать такая возможность, чтобы интеллектуальный агент сам имел активы и участвовал в электронных торгах от своего имени? Создается впечатление, что такие вопросы до сих пор еще полностью не изучены. Насколько известно авторам данной книги, еще ни одной программе не был придан правовой статус как индивидуума, способного участвовать в финансовых операциях; в настоящее время такое решение, по-видимому, было бы неблагодарным. Кроме того, программы еще не рассматриваются как “водители”, когда речь идет о контроле за выполнением правил дорожного движения на реальных автомагистралях. По крайней мере, в Калифорнии не предусмотрено никаких юридических санкций, которые исключали бы возможность превышения автоматизированным транспортным средством скоростных пределов, хотя в случае аварии должен нести ответственность проектировщик механизма управления транспортным средством. Как и в случае технологии клонирования людей, законодателям еще предстоит включить новые разработки в правовое поле.

- Успех искусственного интеллекта может стать началом конца человеческой расы. Почти любая технология, попадая в злонамеренные руки, обнаруживает потенциальные возможности для причинения вреда, но когда речь идет об искусственном интеллекте и робототехнике, возникает новая проблема, связан-

ная с тем, что эти злонамеренные руки могут принадлежать самой технологии. Предупреждения о том, какую опасность несут роботы или роботизированные киборги-гуманоиды, вышедшие из-под контроля, стали сюжетом бесчисленных научно-фантастических произведений. К числу самых ранних примеров таких произведений относятся книга *Frankenstein, or the Modern Prometheus*⁷ Мэри Шелли [1400] и пьеса *R. U. R* Карела Чапека (1921 год), в которых описано, как роботы завоевывают мир. В кинематографе этой теме посвящены фильм “The Terminator” (Терминатор), вышедший на экраны в 1984 году, в котором используются клише “о роботах, завоевывающих мир” и сюжет о путешествии во времени, и фильм “The Matrix” (Матрица), вышедший в 1999 году, в котором объединены сюжеты “о роботах, завоевывающих мир” и “о мозге, растущем в колбе”.

По-видимому, роботы являются главными героями такого большого количества художественных произведений о завоевании мира в основном из-за того, что они воплощают в себе неизвестное, точно так же, как ведьмы и приведения в сказках, которыми пугали людей в более ранние эпохи. Но действительно ли роботы создают более реальную угрозу, чем ведьмы и приведения? Если роботы правильно спроектированы как агенты, которые соблюдают интересы своего владельца, то они не создают таких угроз: роботы, развивающиеся на основе постоянного усовершенствования текущих проектов, будут служить своим хозяевам, а не бороться против них. Люди иногда используют свой интеллект в агрессивных формах, поскольку они обладают некоторыми врожденными агрессивными тенденциями, обусловленными естественным отбором. Но машины, созданные людьми, не нуждаются в чертах врожденной агрессивности, если только сами люди не решат, что они должны быть таковыми. С другой стороны, возможно, что компьютеры добьются своего рода победы над людьми, успешно служа и становясь незаменимыми, так же как и автомобили, которые в определенном смысле завоевали промышленно развитый мир. Рассмотрим один сценарий, заслуживающий дополнительного анализа. И. Дж. Гуд [576] писал в одном из своих произведений следующее.

Дадим определение сверхинтеллектуальной машине как способной намного превзойти во всех областях мыслительной деятельности любого человека, каким бы умным он не был. А поскольку проектирование машин является одним из таких направлений мыслительной деятельности, то сверхинтеллектуальная машина сможет проектировать еще более интеллектуальные машины; это, безусловно, приведет к “взрыву интеллектуальности” и интеллект человека останется далеко позади. Таким образом, первое изобретение сверхинтеллектуальной машины станет последним изобретением, которое когда-либо сможет сделать человек, и то, если только машина будет достаточно любезна, чтобы сообщить человеку, как держать ее под контролем.

Этому “взрыву интеллектуальности” профессор математики и автор научно-фантастических произведений Вернор Виндж присвоил другое название — **технологическое превосходство**; он писал: “В течение тридцати лет люди получают технологические средства для создания сверхчеловеческого интеллекта. Вскоре после этого эра людей закончится” [1542]. Гуд и Виндж (и многие другие) правильно отмечают, что в настоящее время кривая технологического прогресса

⁷ Чарльз Бэббидж в молодости испытал сильные впечатления, читая книгу о приключениях Франкенштейна.

растет экспоненциально (достаточно вспомнить закон Мура). Однако прогноз, что эта кривая будет и дальше подниматься вверх, следуя законам почти бесконечного роста, был бы слишком смелым. До сих пор любая технология развивалась по S-образной кривой, согласно которой экспоненциальный рост в конечном итоге сходит на нет.

Виндж озабочен и обеспокоен грядущим технологическим превосходством, но другие специалисты в области компьютерных наук, занимающиеся прогнозами, радуются его наступлению. Ханс Моравек в своей книге *Robot: Mere Machine to Transcendent Mind* предсказывает, что роботы сравняются по интеллектуальности с человеком за 50 лет, после чего его превзойдут. В своей книге он пишет следующее.

Довольно скоро они станут способными вытеснить нас из жизни. Но я не очень встревожен такой возможностью, поскольку считаю, что будущие машины — это наше потомство, “порождение нашего разума”, созданное по нашему образу и подобию; это мы, но в более развитой форме. Как и биологические дети предыдущих поколений, они воплотят в себе лучшие надежды человечества на долгосрочное будущее. Это побуждает нас предоставить им все возможное содействие, а затем отойти в сторону, после того, как мы больше не в состоянии будем делать что-то полезное [1085].

Рей Курцвейл в своей книге *The Age of Spiritual Machines* [872] предсказывает, что к 2099 году наметится “мощная тенденция к слиянию человеческого мышления с миром машинного интеллекта, который был изначально создан родом человеческим. Больше не будет какого-либо четкого различия между людьми и компьютерами”. Возникло даже новое слово, **трансгуманизм**, для обозначения активного социального движения, которое охватывает тех, кто готовится к такому будущему. Достаточно сказать, что указанные вопросы становятся вызовом для большинства теоретиков морали, которые считают единственно возможным путем развития сохранение существования человека и самого человеческого рода.

Наконец, рассмотрим эту проблему с точки зрения робота. Если роботы обретут сознание, то трактовка их просто как “машин” (т.е. дальнейшее их принижение) может стать аморальной. Роботы сами должны также действовать морально — нам потребуется запрограммировать в них теорию, по которой они смогут судить, что хорошо и что плохо. Проблема прав и обязанностей роботов рассматривалась в произведениях авторов научной фантастики, начиная с Айзека Азимова [45]. В широко известном фильме “A.I.” [1451] Спилберга основой сюжета является рассказ Брайена Олдисса об интеллектуальном роботе, в программу которого была вложена вера в то, что он — человек, поэтому он не мог понять, почему же его когда-то неизбежно покинет владелица-мать. И в самом рассказе, использованном Спилбергом, и в его фильме приведены доводы в пользу того, что должно быть развернуто движение за гражданские права роботов.

26.4. РЕЗЮМЕ

В данной главе рассматривались приведенные ниже темы.

- Философы используют термин **слабый искусственный интеллект** для обозначения гипотезы о том, что машины могут обладать способностью действовать интеллектуально, и термин **сильный искусственный интеллект** — для обозначения

ния гипотезы, что такие машины можно рассматривать как действительно обладающие разумом (а не имитирующие разумную деятельность).

- Алан Тьюринг отверг как некорректный вопрос: “Могут ли машины мыслить?” и заменил его поведенческим тестом. Он сумел предугадать многие возражения против самой возможности появления мыслящих машин. В настоящее время исследователи искусственного интеллекта почти не уделяют тесту Тьюринга внимания, поскольку предпочитают работать над повышением производительности своих систем в решении практических задач, а не над совершенствованием их способности имитировать людей.
- В настоящее время общепризнано, что психические состояния представляют собой состояния мозга.
- Споры за и против сильного искусственного интеллекта еще не закончились. Но лишь немногие из ведущих исследователей искусственного интеллекта считают, что итогом этих дебатов могут стать какие-либо существенные достижения.
- Проблема сознания продолжает оставаться нерешенной загадкой.
- Выявлено шесть потенциальных угроз обществу, создаваемых искусственным интеллектом и связанной с ним технологией. Авторы данной книги пришли к заключению, что некоторые из этих угроз либо являются маловероятными, либо ненамного отличаются по степени опасности от угроз, создаваемых другими, “неинтеллектуальными” технологиями. Лишь одна из этих угроз особо заслуживает дальнейшего анализа; она заключается в том, что с появлением сверхинтеллектуальных машин наступит такое будущее, которое весьма отличается от современности; это может нам не нравиться, но в данном вопросе у нас может не оказаться другого выбора. Указанные соображения неизбежно приводят к выводу, что мы должны тщательно взвешивать свои действия и в дальнейшем заняться анализом возможных последствий исследований искусственного интеллекта, от которых может зависеть будущее человеческой расы.

БИБЛИОГРАФИЧЕСКИЕ И ИСТОРИЧЕСКИЕ ЗАМЕТКИ

Природа разума всегда оставалась неизменной темой философских рассуждений от древних времен до настоящего времени. В своем произведении “Федон” Платон отдельно рассмотрел и отбросил идею о том, что разум может быть “настройщиком” или образцом организации частей тела, т.е. ту точку зрения, которая близка к взглядам приверженцев функционализма в современной философии разума. Вместо этого он решил, что разум должен быть воплощен в бессмертной, нематериальной душе, отделимой от тела и состоящей из другой субстанции, т.е. встал на позиции дуализма. Аристотель различал несколько видов душ (обозначая их греческим словом “психе” — ψυχή) в живых существах, но некоторые из них он описывал в манере функционализма (дополнительные сведения о функционализме Аристотеля можно найти в [1151]).

Известным приверженцем дуалистических взглядов на человеческий разум был Декарт, но по иронии судьбы он оказал огромное историческое влияние на развитие

механицизма и материализма. Он явно рассматривал животных в качестве автоматов и предугадал предложенный Тьюрингом тест, написав в своей работе: “Нет ничего невероятного в том [что машина] может обладать способностью производить различные перестановки слов, чтобы дать достаточно осмысленный ответ на то, что сказано в ее присутствии, так же, как это способен сделать даже самый глупый из людей” [391]. Энергичная защита Декартом точки зрения на то, что животные являются автоматами, фактически позволила будущим философам скорее прийти к выводу о том, что люди также являются автоматами, даже несмотря на то, что сам он не сделал этого шага. В книге *L'Homme Machine* (Человек-машина) [875] Ламетри действительно приведено явное утверждение, что люди являются автоматами.

Приверженцы современной аналитической философии обычно признают материализм (часто в форме **теории идентичности** состояний мозга [39], [1215], согласно которой психические состояния идентичны состояниям мозга). Но в рамках материализма существуют серьезные разногласия по вопросам отношения к функционализму и применения машинной аналогии для человеческого разума, а также по вопросу о том, могут ли машины мыслить в буквальном понимании этого слова. На первых порах многие философы возражали против идей Тьюринга, изложенных в статье “*Computing Machinery and Intelligence*” [1520], например, Шривен [1374] пытался доказать, что бессмысленно даже говорить о том, что машины могут мыслить, на том основании, что в этом предположении искажается сам смысл данного слова. Но к 1963 году Шривен отказался от этих взглядов; см. дополнение к перепечатке его статьи [28]. Специалист в области компьютерных наук Эдсгер Дейкстра сказал, что “вопрос о том, может ли компьютер мыслить, не более интересен, чем вопрос о том, может ли подводная лодка плавать”. А в [480] приведены доводы в пользу того, что тест Тьюринга не нужен для развития искусственного интеллекта.

Функционализм — это философия разума, которая зародилась под влиянием искусственного интеллекта наиболее естественным образом, поэтому критика функционализма часто принимает форму критики искусственного интеллекта (как в случае воззрений философа Сирла). Следуя классификации, используемой Блоком [138], мы можем различить несколько видов функционализма. **Теория функциональной спецификации** [923], [925] представляет собой одну из разновидностей теории идентичности состояний мозга, в которой выбираются состояния мозга, подлежащие идентификации с психическими состояниями на основе их функциональной роли. А **теория идентичности функционального состояния** [1246], [1248] более полно основана на машинной аналогии. В ней психические состояния идентифицируются не с физическими состояниями мозга, а с абстрактными вычислительными состояниями мозга, явно рассматриваемого как вычислительное устройство. Предполагается, что эти абстрактные состояния не зависят от конкретной физической композиции мозга, а это заставляет думать, что теория идентичности функциональных состояний представляет собой одну из форм дуализма!

И теория идентичности состояний мозга, и различные формы функционализма подвергаются атакам со стороны философов, утверждающих, что в этих взглядах не учитывается качественный аспект психических состояний или “аспект их подобия” [1110]. С другой стороны, Сирл сосредоточивается не на этом вопросе, а на том, что функционализм якобы не позволяет учитывать целенаправленность [1376], [1377], [1379]. В [259] приведено опровержение критических взглядов обоих этих типов.

Отличием ~~э~~ **элиминирующего материализма** [258], [1301] от всех других ведущих теорий в философии разума является то, что в нем не предпринимаются попытки обосновать правильность представлений “народной психологии” или обыденных представлений о разуме; вместо этого подобные взгляды отвергаются как ложные и предпринимается попытка заменить их чисто научной теорией разума. В принципе эта научная теория могла бы быть лучше развита в рамках классического искусственного интеллекта, но на практике приверженцы элиминирующего материализма склонны к проведению исследований в области неврологии и нейронных сетей [260] на том основании, что классический искусственный интеллект, особенно исследования в области “представления знаний”, описанные в главе 10, как правило, исходят из истинности “народной психологии”. Хотя точка зрения, основанная на “целенаправленной позиции” [387], может рассматриваться как соответствующая принципам функционализма, ее, возможно, следует вместо этого считать одной из форм элиминирующего материализма, поскольку предполагается, что принятие “целенаправленной позиции” не отражает каких-либо объективных свойств агента, по отношению к которым занята эта позиция. Следует также отметить такую возможность, что позиция элиминирующего материализма будет принята в отношении одних аспектов мыслительной деятельности, тогда как в отношении других будет принят какой-то другой подход. Например, Деннет [388] выдвигает гораздо более строгие требования по устранению из сферы анализа (в рамках элиминирующего материализма) понятия *качества*, чем понятия *целенаправленности*.

Ссылки на литературу, содержащую основные критические замечания в адрес слабого искусственного интеллекта, в основном приведены в данной главе. Хотя в эпоху, наступившую после бума в области нейронных сетей, стало модно высмеивать символические подходы, не все философы критически относятся к достижениям GOFAI. Напротив, некоторые из них являются его горячими сторонниками и даже практиками. Зенон Пилишин [1250] утверждает, что процесс познания можно лучше всего понять с помощью вычислительной модели, не только в принципе, но и в ходе проведения современных исследований. Он специально занимался опровержением критических замечаний Дрейфуса в адрес вычислительной модели человеческого познания [1249]. Гильберт Харман [621], анализируя процесс пересмотра воззрений, провел аналогии с исследованиями искусственного интеллекта на базе системы поддержки истинности. Майкл Братман применил свою модель человеческой психологии “убеждение—желание—намерение” [176] к исследованиям искусственного интеллекта по планированию [177]. Крайний приверженец строгого искусственного интеллекта, Арон Сломан [1432, с. xiii], даже назвал “расистскими” взгляды Джозефа Вейценбаума [1566] на то, что гипотетические интеллектуальные машины не следует рассматривать как личности.

Философская литература по проблемам разума, мозга и близким к этому темам очень обширна; кроме того, эта литература часто является трудной для чтения теми, кто не обладает надлежащей подготовкой в области терминологии и используемых методов аргументирования. Исключительно авторитетным и очень полезным помощником в этом процессе может стать книга *Encyclopedia of Philosophy* (Энциклопедия философии) [431]. Более краткой и доступной книгой является *The Cambridge Dictionary of Philosophy* (Кембриджский словарь по философии) [48], но и в нем основные статьи (например, с описанием “философии разума”) все еще превышают по объему десятки страниц. Вопросы философии разума, а также биологии и психологии разума

рассматриваются в книге *MIT Encyclopedia of Cognitive Science* [1599]. К числу общих сборников статей о философии разума, в которых представлены различные точки зрения на проблемы, связанные с искусственным интеллектом, включая функционализм, относятся *Materialism and the Mind-Body Problem* [1309] и *Readings in the Philosophy of Psychology*, том 1 [138]. Биро и Шахан представили сборник статей [131], посвященный всем доводам за и против функционализма. К антологиям статей, в большей степени касающихся отношений между философией и искусственным интеллектом, относятся *Minds and Machines* [28], *Philosophical Perspectives in Artificial Intelligence* [1288], *Mind Design* [630] и *The Philosophy of Artificial Intelligence* [146]. Имеется также несколько общих введений в философский “вопрос об искусственном интеллекте” [145], [146], [293], [631]. Основным журналом, который публикует материалы философских и научных дебатов об искусственном интеллекте и неврологии, является *The Behavioral and Brain Sciences* (сокращенно *BBS*). Вопросы этики и ответственности в искусственном интеллекте рассматриваются в таких журналах, как *AI and Society*, *Law, Computers and Artificial Intelligence* и *Artificial Intelligence and Law*.

УПРАЖНЕНИЯ

- 26.1. Пройдите по составленному Тьюрингом списку предполагаемых действий, на которые “не способны” машины, и укажите, какие из них удалось осуществить, какие осуществимы в принципе с помощью программ и какие все еще находятся под вопросом, поскольку для них требуются сознательные психические состояния.
- 26.2. Является ли опровержение довода с китайской комнатой убедительным доказательством того, что должным образом запрограммированные компьютеры имеют психические состояния? Обязательно ли принятие этого довода означает согласие с тем, что компьютеры не могут иметь психических состояний?
- 26.3. Пользуясь доводом с протезом мозга, важно предусмотреть способность восстановить мозг субъекта до нормального состояния, так, чтобы внешне он вел себя, будто над ним не проводилась какая-либо операция. Будет ли обоснованным возражение скептика о том, что для этого потребуется обновление нейрофизиологических свойств нейронов, которые относятся к сознательному опыту, в отличие от тех свойств, которые касаются функционального поведения нейронов?
- 26.4. Найдите и проанализируйте в популярном издании одну или несколько статей на ту тему, что создание искусственного интеллекта является невозможным.
- 26.5. Попытайтесь составить определения терминов “интеллект”, “мышление” и “сознание”. Проанализируйте некоторые возможные возражения против ваших определений.
- 26.6. Проанализируйте потенциальные угрозы обществу со стороны технологии искусственного интеллекта. Какие угрозы являются наиболее серьезными и как им можно противостоять? В каком соотношении они находятся с потенциальными выгодами?

- 26.7. Каковыми являются потенциальные угрозы со стороны технологии искусственного интеллекта в сравнении с угрозами со стороны других компьютерных технологий, а также со стороны ядерных, био- и нано- технологий?
- 26.8. Одни критики искусственного интеллекта утверждают, что его создать невозможно, а другие считают, что очень даже возможно и что сверхинтеллектуальные машины представляют собой угрозу. Какое из этих утверждений вы считаете наиболее вероятным? Будут ли противоречивыми взгляды тех, кто занимает одновременно обе позиции?

27 НАСТОЯЩЕЕ И БУДУЩЕЕ ИСКУССТВЕННОГО ИНТЕЛЛЕКТА

В данной главе мы подводим итог тому, где мы находимся и куда движемся; это следует сделать, прежде чем приступить к дальнейшему изложению ее темы.

В части I предложен единый подход к проблематике искусственного интеллекта — как к проектированию рациональных агентов. В ней было показано, что сложность задачи проектирования зависит от того, какие акты восприятия и действия доступны для агента, каким целям должно удовлетворять поведение агента, а также от характера среды. Возможно применение целого ряда различных проектов агентов, начиная от простых рефлексных агентов и заканчивая агентами, основанными на знаниях, полностью способными к самостоятельному функционированию. Более того, компоненты этих проектов могут иметь целый ряд различных реализаций, таких как логические, вероятностные или “нейронные”. А в главах, которые следуют за этой частью, представлены принципы, на основании которых действуют эти компоненты.

Как описано выше, в последнее время достигнуты поразительные успехи в научном понимании проблемы и в обеспечении технологических возможностей как с точки зрения разработки проектов, так и с точки зрения реализации компонентов агентов. В данной главе мы отвлечемся от деталей и попытаемся найти ответ на вопрос: ☞ “Приведет ли весь этот прогресс к созданию интеллектуального агента общего назначения, способного функционировать в самых различных вариантах среды?” В разделе 27.1 рассматриваются компоненты интеллектуального агента для определения того, что известно и чего еще недостает. В разделе 27.2 решается такая же задача применительно к общей архитектуре агента. В разделе 27.3 предпринимается попытка найти ответ на вопрос о том, является ли, прежде всего, правильной сама цель создания “проекта рационального агента” (ответ на данный вопрос должен быть таков: “Фактически дело обстоит иначе, но на данный момент эта цель являет-

ся вполне приемлемой"). Наконец, в разделе 27.4 показано, к каким последствиям приведет достижение нами успеха в своих начинаниях.

27.1. КОМПОНЕНТЫ АГЕНТА

В главе 2 представлены разнообразные проекты агентов и их компоненты. Но в настоящей главе, для того чтобы можно было сузить рамки обсуждения, будет рассматриваться проект агента, действующего с учетом полезности, который еще раз показан на рис. 27.1. Это — наиболее общий из рассматриваемых нами проектов агентов; в этой главе речь также пойдет о его дополнении средствами обучения, как показано на рис. 2.7.

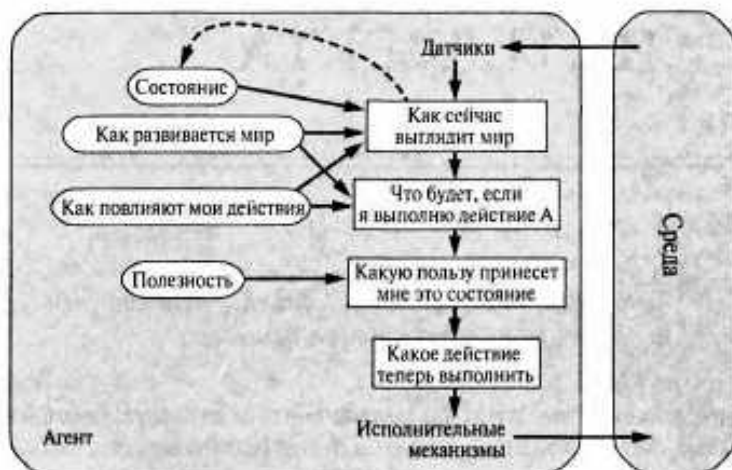


Рис. 27.1. Проект, основанный на модели агента, действующего с учетом полезности, который впервые представлен на рис. 2.6

- Взаимодействие со средой с помощью датчиков и исполнительных механизмов.** Этот аспект оставался очевидным для всех слабым пунктом на протяжении большей части истории развития искусственного интеллекта. Если не считать небольшого числа исключений, вызвавших всеобщее восхищение, системы искусственного интеллекта создавались таким образом, что людям приходилось вручную подавать в них входные данные и интерпретировать выходные, поскольку робототехнические системы, предназначенные для решения задач низкого уровня, на которые опирались бы высокоуровневые компоненты формирования рассуждений и планирования, в основном отсутствовали. Такая ситуация отчасти была обусловлена тем, что для обеспечения функционирования настоящих роботов даже в самой узкой области требовались существенные финансовые расходы и большие затраты труда проектировщиков. Такая ситуация быстро изменяется в последние годы в связи с появлением готовых программируемых роботов, таких как роботы с четырьмя опорными конечностями (см. рис. 25.4, б). Эти роботы, в свою очередь, созданы благодаря появлению небольших, недорогих телекамер CCD (Charge Coupled Device — прибор с за-

рядовой связью) с высоким разрешением, а также компактных, надежных электрических приводов. Технология MEMS (Micro-ElectroMechanical System — микроскопические электромеханические системы) позволила создать миниатюрные акселерометры и гироскопы, а в настоящее время в ее рамках создаются исполнительные механизмы, способные, например, приводить в действие искусственное летающее насекомое. (Существует также возможность объединять миллионы исполнительных механизмов MEMS для получения очень мощных макроскопических исполнительных механизмов.) Поэтому, что касается вариантов физической среды, то больше нет реальных причин, оправдывающих то положение, в котором находятся системы искусственного интеллекта. Кроме того, стала доступной полностью новая среда — Internet.


- **Слежение за состоянием мира.** Это — одна из основных способностей, которой должен обладать интеллектуальный агент. Для этого требуется и восприятие, и обновление внутренних представлений. В главе 7 описаны методы слежения за миром, представленные в форме пропозициональной логики; в главе 10 они расширены до логики первого порядка, а в главе 15 представлены алгоритмы **фильтрации** для слежения за неопределенными вариантами среды. Эти инструментальные средства фильтрации вступают в действие, когда приходится сталкиваться с реальными (поэтому далекими от идеала) результатами восприятия. Современные алгоритмы фильтрации и восприятия могут комбинироваться для успешного выполнения заданий по составлению сообщений в виде предикатов низкого уровня, таких как “на столе стоит чашка”, но еще многое предстоит сделать, прежде чем с помощью этих алгоритмов можно будет составить отчет, например, о том, что “доктор Рассел пьет чай с доктором Норвигом”. Еще одна проблема состоит в том, что алгоритмы приближенной фильтрации, хотя и могут действовать в весьма обширной среде, остаются по сути пропозициональными, поэтому, как и пропозициональная логика, не позволяют явно представлять объекты и отношения. В главе 14 описано, как можно применить в сочетании теорию вероятностей и логику первого порядка для решения этой задачи; можно рассчитывать на то, что применение этих идей для слежения за сложными вариантами среды со временем позволит добиться огромных преимуществ. Кстати, как только речь заходит об объектах в неопределенной среде, нам приходится сталкиваться с **неопределенностью идентичности**, поскольку часто неизвестно, не потерял ли из виду тот объект, за которым мы начинали следить. Эта проблема в системах искусственного интеллекта, основанных на логике, почти всегда игнорировалась, поскольку в основном предполагалось, что результаты восприятия включают константные символы, которые однозначно обозначают те или иные объекты.
- **Проектирование, оценка и выбор будущих способов действий.** При решении этой задачи требования к представлению основных знаний остаются такими же, как и при решении задачи слежения за миром; трудности состоят главным образом в том, что приходится сталкиваться с проявлениями действий (например, связанных с проведением беседы или совместного чаепития), которые в конечном итоге состоят из тысяч или миллионов примитивных шагов, выполняемых реальным агентом. Вообще говоря, люди осуществляют такое сложное поведение исключительно благодаря тому, что действуют в рамках

иерархической структуры поведенческих актов. В некоторых из алгоритмов планирования, приведенных в главе 12, используются иерархические представления и представления в логике первого порядка, позволяющие справляться с проблемами реальных масштабов; с другой стороны, в алгоритмах принятия решений в условиях неопределенности, приведенных в главе 17, по существу используются такие же идеи, как и в алгоритмах поиска с учетом состояния, рассматриваемых в главе 3. В этой области необходимо выполнить еще очень большой объем работы, возможно, на основе новейших достижений в области **иерархического обучения с подкреплением**.

- **Полезность как способ выражения предпочтений.** Вообще говоря, принцип, согласно которому рациональные решения должны быть основаны на максимизации ожидаемой полезности, является полностью общим и позволяет избежать многих проблем, связанных с подходами, основанными исключительно на достижении цели, таких как конфликтующие цели и ненадежные результаты. Однако до сих пор еще очень мало сделано в области создания реальных функций полезности. Достаточно представить себе, например, в какой сложной сети взаимодействующих предпочтений должен разбираться агент, действующий в качестве ассистента-делопроизводителя для чиновника. Как оказалось, задача декомпозиции предпочтений по сложным состояниям, подобная тому, как осуществляется декомпозиция убеждений по сложным состояниям в байесовских сетях, является весьма трудноразрешимой. Одна из причин этого может состоять в том, что распределение предпочтений по состояниям фактически компилируется из предпочтений, распределенных по историям состояний, которые описываются с помощью **функций вознаграждения** (см. главу 17). Даже если функция вознаграждения является простой, соответствующая функция полезности может оказаться очень сложной. Это означает, что мы должны рассматривать задачу инженерии знаний для функций вознаграждения, которая должна стать способом информирования разрабатываемых агентов о том, какие к ним предъявляются требования, как очень серьезную.
- **Обучение.** В главах 18–20 описано, как может быть сформулирована задача обучения агента в виде задачи определения с помощью индуктивного обучения (контролируемого, неконтролируемого или основанного на подкреплении) тех функций, которые лежат в основе различных компонентов агента. Были разработаны очень мощные логические и статистические методы, позволяющие справляться с весьма значительными проблемами, часто достигаящие или превосходящие возможности человека по идентификации предсказательных шаблонов, определенных в заданном словаре. С другой стороны, в области машинного обучения достигнуты лишь весьма небольшие успехи в решении важной проблемы формирования новых представлений на уровнях абстракции, более высоких по сравнению с входным словарем. Например, как может автономный робот выработать полезные предикаты, такие как *Office* и *Cafe*, если они не будут предоставлены ему разработчиком? Аналогичные соображения распространяются и на проблему определения с помощью обучения способа поведения; например, участие в чаепитии *HavingACupOfTea* — это важное действие высокого уровня, но как ввести его в библиотеку действий, которая первоначально содержит гораздо более

простые действия, такие как *RaiseArm* (Поднять руку) и *Swallow* (Сделать глоток)? Если мы не поймем специфику таких проблем, то столкнемся с утомительной задачей построения больших баз обыденных знаний вручную.

27.2. АРХИТЕКТУРЫ АГЕНТОВ

Возникает резонный вопрос: “Какие архитектуры агентов, описанные в главе 2, должны использоваться в агентах?” Ответом является: “Все архитектуры!” Выше было показано, что рефлексные реакции требуются в тех ситуациях, в которых существенным является фактор времени, а с другой стороны, рассуждения, основанные на знаниях, позволяют агенту планировать наперед. Полноценный агент должен быть способным выполнять и то и другое с использованием  **гибридной архитектуры**. Одним из важных свойств гибридных архитектур является то, что границы между различными компонентами, обеспечивающими принятие решений, не постоянны. Например, в процессе **компиляции** декларативная информация, полученная на уровне формирования рассуждений, последовательно преобразуется во все более эффективные представления, что позволяет в конечном итоге достичь рефлексного уровня, как показано на рис. 27.2. (В этом состоит цель обучения на основе объяснения, как описано в главе 19.) Точно такую же структуру имеют такие архитектуры агентов, как Soar [880] и Theo [1063]. После решения каждой задачи с помощью явного формирования рассуждений эти агенты сохраняют обобщенную версию полученного решения для его использования в рефлексном компоненте. Менее изученной проблемой является проблема осуществления процесса, обратного указанному, — после изменения среды рефлекссы, усвоенные в результате обучения, могут оказаться больше не подходящими, и агенту может потребоваться вернуться на уровень формирования рассуждений, для того чтобы выработать новые способы поведения.

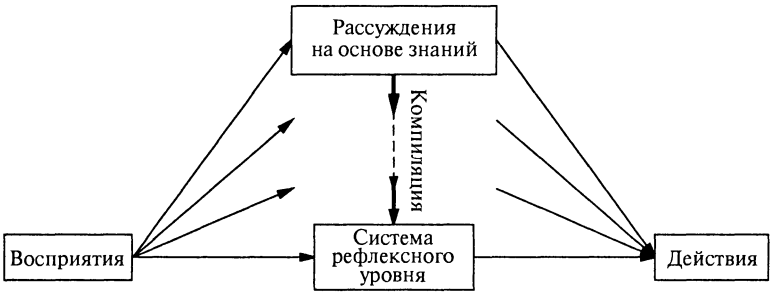


Рис. 27.2. Компиляция служит для преобразования результатов принятия решений на основе рассуждений в более эффективные рефлексивные механизмы

Агентам могут также потребоваться способы, позволяющие управлять своими собственными процессами формирования рассуждений. Они должны быть способными прекратить размышления, когда потребуются действия, а также должны уметь использовать время, отведенное на рассуждения, чтобы выполнить наиболее продуктивные вычисления. Например, агент-водитель такси, который обнаружил впереди картину дорожного происшествия, должен решить за долю секунды, следует ли ему затормозить или объехать то место, где случилось происшествие. Кроме того,

данный агент должен также потратить лишь долю секунды на размышление о наиболее важных в этой ситуации вопросах, например, нет ли движения на полосах слева и справа и нет ли непосредственно сзади него большого грузовика, но не задумываться над тем, что резкий маневр увеличит износ и стирание шин автомобиля или что ему давно нужно было найти очередного пассажира. Исследование таких проблем осуществляется главным образом в рамках направления **искусственного интеллекта реального времени**. По мере того как системы искусственного интеллекта проникают во все более сложные проблемные области, все решаемые задачи становятся задачами реального времени, поскольку агенту никогда больше не отводится достаточно времени для точного решения задачи принятия решений.

Очевидно, что становится насущной потребность в методах, которые позволяют действовать в более общих ситуациях принятия решений. В последние годы появились два перспективных метода. В первом из них предусматривается использование **алгоритмов с отсечением по времени** [357], [682]. Алгоритмом с отсечением по времени называется алгоритм, качество выходных данных которого неизменно улучшается во времени, поэтому он всегда готов предоставить приемлемое решение, когда бы ни была прервана его работа. Такие алгоритмы действуют под управлением метауровневой процедуры принятия решений, которая оценивает, стоит ли выполнять дальнейшие вычисления. Простым примером алгоритма с отсечением по времени является поиск с итеративным углублением в задачах ведения игр. Могут также быть созданы более сложные системы, состоящие из многих таких алгоритмов, действующих вместе [1647]. Вторым методом являются **метарассуждения на основе теории решений** [683], [687], [1332]. В этом методе применяется теория ценности информации (см. главу 16) для выбора вычислений. Ценность вычислений зависит и от их стоимости (с точки зрения того, что действие не проводится, пока они осуществляются) и от их преимуществ (измеряемых с учетом того, насколько повысилось качество решения). Методы формирования метарассуждений могут использоваться для проектирования лучших алгоритмов поиска и для обеспечения гарантий того, что алгоритмы будут обладать вневременным свойством. Безусловно, подход на основе метарассуждений является дорогостоящим, а методы компиляции могут применяться таким образом, чтобы издержки были малы по сравнению со стоимостями контролируемых вычислений.

Тем не менее метарассуждения представляют собой лишь один из аспектов общей **рефлексивной архитектуры**, т.е. архитектуры, позволяющей формировать рассуждения о вычислительных сущностях и действиях, возникающих в самой архитектуре. Теоретическую основу для рефлексивных архитектур можно заложить, определив совместное пространство состояний, складывающееся из состояний среды и вычислительного состояния самого агента. Могут быть спроектированы алгоритмы принятия решений и обучения, которые применяются к этому совместному пространству состояний и поэтому способствуют реализации и совершенствованию вычислительной деятельности агента. Мы надеемся, что в конечном итоге такие алгоритмы, предназначенные для решения узко конкретных задач, как альфа-бета поиск и обратный логический вывод, исчезнут из систем искусственного интеллекта и будут заменены общими методами, которые направляют вычисления агента в сторону эффективного формирования высококачественных решений.

27.3. ОЦЕНКА ПРАВИЛЬНОСТИ ВЫБРАННОГО НАПРАВЛЕНИЯ

В предыдущем разделе были перечислены многие достижения и многие возможности для дальнейшего прогресса. Но в каком направлении идет все это развитие? Дрейфус [415] проводит аналогию с попыткой достать до луны, взбираясь на дерево; в ходе этого наблюдается постоянный прогресс до тех пор, пока не будет достигнута вершина дерева. В этом разделе мы рассмотрим, напоминает ли текущий путь развития искусственного интеллекта подъем по стволу дерева или взлет ракеты.

В главе 1 было указано, что наша цель состоит в создании агентов, которые действуют рационально, но в той же главе было также указано следующее:

...задача достижения идеальной рациональности, при которой всегда выполняются правильные действия, не осуществима. Дело в том, что при этом предъявляются слишком высокие требования к вычислительным ресурсам. Но в основной части данной книги применяется рабочая гипотеза, согласно которой идеальная рациональность является хорошей отправной точкой для анализа.

Теперь настало время определить, в чем именно состоит цель искусственного интеллекта. Мы стремимся создавать агентов, но какой спецификацией следует при этом руководствоваться? Ниже описаны четыре возможных варианта.

- **Идеальная рациональность.** Идеально рациональный агент в каждое мгновение действует таким образом, чтобы максимизировать свою ожидаемую полезность с использованием информации, полученной им из среды. Но было показано, что вычисления, необходимые для достижения идеальной рациональности, в большинстве вариантов среды требуют слишком больших затрат времени, поэтому задача обеспечения идеальной рациональности не является реалистичной.
- **Вычислительная рациональность.** Это понятие рациональности использовалось нами неявно при проектировании логических агентов и агентов, основанных на принятии решений. Вычислительно рациональный агент в конечном итоге возвращает то, что рассматривалось как рациональный выбор в начале этапа формирования им рассуждений. Такое свойство может представлять интерес, только если система используется лишь в демонстрационных целях, а в большинстве вариантов среды правильный ответ теряет свою ценность, если он не был своевременным. На практике проектировщики систем искусственного интеллекта вынуждены искать компромисс между требованиями по обеспечению качества решений и требованиями по уменьшению затрат времени на получение приемлемой общей производительности; к сожалению, теоретические основы вычислительной рациональности не позволяют предложить достаточно обоснованного способа выработки таких компромиссов.
- **Ограниченная рациональность.** Герберт Саймон [1415] отверг идею идеальной (или даже приближенно идеальной) рациональности и предложил использовать понятие ограниченной рациональности — описательную теорию принятия решений реальными агентами. Ниже приведена цитата из его работы.

Способность человеческого разума формулировать и решать сложные задачи слишком мала по сравнению с масштабами задач, для которых требуется искать решение, чтобы осуществлять объективно рациональное поведение в реальном мире, или хотя бы добиваться приемлемого приближения к такой объективной рациональности.

Саймон выдвинул предложение, что принцип ограниченной рациональности главным образом основан на принципе **непритязательности** (satisficing), т.е. на том, что проведение рассуждений следует осуществлять достаточно долго лишь для того, чтобы предложить “вполне приемлемый” ответ. За указанную выше работу Саймон получил Нобелевскую премию по экономике, а позднее выпустил книгу, в которой подробно осветил все свои идеи [1418]. По-видимому, понятие непритязательности может служить полезной моделью человеческого поведения во многих случаях. Но оно не является формальной спецификацией для интеллектуальных агентов, поскольку в теории Саймона не дано определение “вполне приемлемого” ответа. Кроме того, принцип непритязательности, по-видимому, является лишь одним из широкого спектра методов, используемых для осуществления действий в условиях ограниченных ресурсов.

- **Ограниченная оптимальность** (Bounded Optimality — BO). Ограниченно оптимальный агент действует настолько хорошо, насколько это возможно, с учетом его вычислительных ресурсов. Это означает, что ожидаемая полезность программы агента для ограниченного оптимального агента является по меньшей мере такой же высокой, как и ожидаемая полезность любой другой агентской программы, работающей на том же компьютере.

По-видимому, наилучшие перспективы создания мощного теоретического фундамента для искусственного интеллекта открывает только одна из этих четырех возможностей — ограниченная оптимальность. Преимущество связанного с ней подхода состоит в том, что он является осуществимым, — всегда можно найти по меньшей мере одну наилучшую программу, а таким свойством не обладает подход с идеальной рациональностью. Ограниченно оптимальные агенты действительно применимы в реальном мире, тогда как вычислительно рациональные агенты обычно неприменимы, а агенты, действующие по принципу непритязательности, могут оказаться применимыми или нет, в зависимости от их собственных конструктивных особенностей.

Традиционным подходом в искусственном интеллекте было то, что нужно начинать с вычислительной рациональности, а затем вырабатывать компромиссы с учетом ресурсных ограничений. Если проблемы, связанные с применением ограничений, не столь существенны, то можно надеяться на создание окончательного проекта, аналогичного проекту ограниченно оптимального агента. Но, по мере того как ресурсные ограничения становятся все более важными (например, по мере усложнения среды), может оказаться так, что два проекта станут весьма несхожими. А в теории ограниченной оптимальности эти ограничения могут учитываться в рамках целостного подхода.

До сих пор объем знаний в области ограниченной оптимальности остается не таким уж значительным. Известно, что могут быть созданы ограниченно оптимальные программы для очень простых машин и для довольно лимитированных вариантов среды [445], [1330], но еще нет полного представления о том, какими должны быть программы ВО для больших компьютеров общего назначения, применяемых в сложных вариантах среды. Если теория ограниченной оптимальности будет носить конструктивный характер, то можно рассчитывать на получение проектов ограниченно оптимальных программ, которые не слишком сильно зависят от устройства используемого компьютера. Научные исследования стали бы весьма затруднительными, если бы увеличение объема памяти гигабайтового компьютера на несколько килобайтов привело к существенному изменению программы ВО. Одним из способов обеспечения того, чтобы это не могло

случиться, может служить небольшое ослабление критериев ограниченной оптимальности. По аналогии с понятием асимптотической сложности (приложение А) можно определить понятие **асимптотической ограниченной оптимальности** (Asymptotic Bounded Optimality — АВО), как описано ниже [1329]. Предположим, что программа P является ограниченно оптимальной для компьютера M в классе вариантов среды E , тогда как сложность вариантов среды в E не ограничена. В таком случае программа P' обладает свойством АВО для M в E , если она может превзойти по производительности программу P , работая на компьютере kM , который в k раз быстрее (или крупнее) по сравнению с M . За исключением предельных значений k было бы достаточно иметь программу, обладающую свойством АВО, для нетривиальной среды в нетривиальной архитектуре. Было бы мало смысла затрачивать невероятные усилия на поиск программ ВО, а не АВО, поскольку все равно размеры и скорость доступных компьютеров увеличиваются на постоянный коэффициент через фиксированные промежутки времени, в связи с появлением каждого нового поколения этих устройств.

Можно рискнуть предположить, что программы ВО или АВО для мощных компьютеров, действующих в сложных вариантах среды, не обязательно должны иметь простую, изящную структуру. Выше уже было показано, что для искусственного интеллекта общего назначения требуются некоторые рефлексивные способности и некоторые способности к формированию рассуждений, целый ряд форм представления знаний и принятия решений, механизмы обучения и компиляции для всех этих форм, методы управления процессом формирования рассуждений и большой запас знаний в данной конкретной области. Ограниченно оптимальный агент должен адаптироваться к той среде, в которой он находится сам, с тем чтобы в конечном итоге его внутренняя организация соответствовала возможностям оптимизации, характерным для данной конкретной среды. Можно рассчитывать лишь на указанную возможность, а такой ход развития аналогичен пути, по которому развивались гоночные автомобили с ограничениями на мощность, пока наконец не были созданы чрезвычайно сложные, но весьма эффективные проекты. По мнению авторов, наука искусственного интеллекта, основанная на понятии ограниченной оптимальности, будет способствовать интенсивному исследованию процессов, позволяющих путем последовательных итераций создавать агентские программы с ограниченной оптимальностью и, возможно, меньше сосредоточиваться на анализе того, как именно устроены создаваемые при этом программы, пусть даже не такие изящные.

Подводя итог, можно отметить, что проведение разработки понятия ограниченной оптимальности было предложено в качестве формальной задачи для исследований по искусственному интеллекту, которая не только хорошо определена, но и осуществима. Ограниченная оптимальность определяет оптимальные программы, но не оптимальные действия. А действия в конечном итоге вырабатываются программами и с помощью программ, которые полностью зависят от замысла проектировщика.

27.4. ПЕРСПЕКТИВЫ РАЗВИТИЯ ИСКУССТВЕННОГО ИНТЕЛЛЕКТА

Дэвид Лодж в своем романе *Small World* [943] об академическом мире литературной критики описал сцену, в которой главный герой вызывает замешательство среди

группы выдающихся, но несогласных друг с другом теоретиков литературы, задав им следующий вопрос: “Что было бы, если бы вы оказались правы?” Ни один из теоретиков, по-видимому, еще не задумывался над этим вопросом раньше, поскольку ведение споров по поводу неопровержимых теорий — бессмысленное занятие. Аналогичное замешательство можно вызвать, задав исследователям в области искусственного интеллекта вопрос: “Что было бы, если бы вам удалось достичь своей цели?” Ведь искусственный интеллект демонстрирует замечательные достижения, а интеллектуальные компьютеры, безусловно, более полезны, чем неинтеллектуальные компьютеры, так о чем же беспокоиться?

Как было указано в разделе 26.3, необходимо рассмотреть некоторые этические проблемы. Интеллектуальные компьютеры являются более мощными, но будет ли эта мощь использоваться во благо или во зло? Те, кто посвящают свою жизнь разработкам в области искусственного интеллекта, ответственны за то, чтобы влияние их работы было положительным. Широта этого влияния зависит от степени успеха искусственного интеллекта. Даже первые скромные успехи в области искусственного интеллекта повлияли на то, как осуществляются преподавание компьютерных наук [1459] и разработка программного обеспечения. Благодаря искусственному интеллекту удалось создать принципиально новые приложения, такие как системы распознавания речи, системы управления запасами, интеллектуальные системы наружного наблюдения, роботы и машины поиска.

Авторы считают, что достижение в искусственном интеллекте успехов среднего уровня окажет влияние на повседневную жизнь всех слоев населения во всем мире. До сих пор такого рода всепроникающее воздействие на общество смогли оказать лишь компьютеризированные сети связи, такие как сеть сотовой телефонной связи и Internet, а искусственный интеллект оставался в стороне. Вполне можно представить себе, что действительно полезные персональные ассистенты для офиса или дома окажут большое положительное воздействие на повышение качества повседневной жизни, хотя они в краткосрочной перспективе и могут вызвать некоторые экономические неурядицы. Кроме того, технологические возможности, открывающиеся на этом уровне, могут быть также применены для создания автономного оружия, появление которого многие считают нежелательным.

Наконец, кажется вполне вероятным, что крупномасштабный успех в создании искусственного интеллекта (появление интеллекта на уровне человека и превосходящего его) повлияет на существование большинства представителей рода человеческого. Изменится сам характер нашей работы и развлечений, так же как и наши представления об интеллекте, сознании и будущей судьбе человечества. На этом уровне системы искусственного интеллекта могут создать более непосредственную угрозу самоопределению, свободе и даже выживанию людей. По этим причинам нельзя рассматривать исследования в области искусственного интеллекта в отрыве от их этических последствий.

Каким мы видим будущее? Авторы научно-фантастических романов, по-видимому, чаще публикуют пессимистические, а не оптимистические сценарии грядущего, возможно, потому, что это позволяет сочинять более увлекательные сюжеты. Но создается впечатление, что искусственный интеллект пока развивается по такому же принципу, как и другие революционные технологии (типографское дело, инженерное оборудование, воздухоплавание, телефония и т.д.), отрицательные последствия внедрения которых перевешиваются положительными результатами.

В заключение следует отметить, что за свою короткую историю искусственный интеллект добился существенного прогресса, но последнее утверждение из эссе Алана Тьюринга *Computing Machinery and Intelligence* продолжает оставаться неоспоримым и в наши дни.

Мы способны заглянуть вперед лишь на очень короткое расстояние, но все равно можем увидеть, как много еще предстоит сделать.

А МАТЕМАТИЧЕСКИЕ ОСНОВЫ

А.1. АНАЛИЗ СЛОЖНОСТИ И СИСТЕМА ОБОЗНАЧЕНИЙ $O()$

Специалистам в области компьютерных наук часто приходится решать задачу сравнения алгоритмов для определения того, насколько быстро они действуют или сколько памяти для них требуется. Для решения этой задачи предусмотрены два подхода. Первым из них является \approx **применение эталонных тестов** — прогон реализующих алгоритмы программ на компьютере и измерение быстродействия в секундах и потребления памяти в байтах. Верно, что в конечном итоге нас действительно интересуют именно такие практические характеристики, но эталонное тестирование может оказаться неприемлемым потому, что оно слишком узко направлено, — в нем измеряется производительность конкретной программы, написанной на конкретном языке, функционирующей на конкретном компьютере с конкретным компилятором и с конкретными входными данными. К тому же на основании единственного результата, полученного с помощью эталонного тестирования, трудно предсказать, насколько успешно этот алгоритм будет действовать в случае использования другого компилятора, компьютера или набора данных.

Асимптотический анализ

Второй подход опирается на математический \approx **анализ алгоритмов**, независимый от конкретной реализации и входных данных. Рассмотрим этот подход на приведенном ниже примере программы, которая вычисляет сумму последовательности чисел (листинг А.1).

Листинг А.1. Программа вычисления суммы последовательности чисел

```
function Summation(sequence) returns сумма sum
    sum  $\leftarrow$  0
    for i  $\leftarrow$  1 to Length(sequence)
        sum  $\leftarrow$  sum + sequence[i]
    return sum
```

Первый этап анализа состоит в том, что создается определенное абстрактное представление входных данных, позволяющее найти какой-то параметр (или параметры), который характеризует объем входных данных. В рассматриваемом примере объем входных данных можно охарактеризовать с помощью такого параметра, как длина последовательности, которая будет обозначаться как n . На втором этапе необходимо определить абстрактное представление реализации и найти какой-то критерий, отражающий продолжительность прогона алгоритма, но не привязанный к конкретному компилятору или компьютеру. Применительно к программе *Summation* этим критерием может служить количество строк выполняемого кода; кроме того, данный критерий может быть более детализированным и измеряющим количество сложений, присваиваний, обращений к элементам массивов, а также ветвлений, выполняемых в этом алгоритме. В любом случае будет получена характеристика общего количества шагов, выполняемых алгоритмом, как функция от объема входных данных. Обозначим эту характеристику как $T(n)$. Если за основу берется количество строк кода, то в данном примере $T(n) = 2n + 2$.

Если бы все программы были такими же простыми, как *Summation*, то область анализа алгоритмов не заслуживала бы названия научной. Но исследования в этой области существенно усложняются из-за наличия двух проблем. Первая проблема заключается в том, что редко удается найти параметр, подобный $T(n)$, который полностью характеризует количество шагов, выполняемых в алгоритме. Вместо этого обычно можно самое большее вычислить этот показатель для наихудшего случая $T_{\text{worst}}(n)$ или для среднего случая $T_{\text{avg}}(n)$. А для вычисления среднего требуется, чтобы аналитик принял какие-то обоснованные предположения по распределению наборов входных данных.

Вторая проблема состоит в том, что алгоритмы обычно не поддаются точному анализу. В этом случае приходится прибегать к аппроксимации и использовать, например, такую формулировку, что быстроедействие алгоритма *Summation* характеризуется величиной $O(n)$. Это означает, что быстроедействие алгоритма измеряется величиной, пропорциональной n , возможно, за исключением нескольких небольших значений n . Более формально это определение можно представить с помощью следующей формулы:

$T(n)$ равно $O(f(n))$, если $T(n) \leq kf(n)$ для некоторого k , при всех $n > n_0$

Перейдя к использованию системы обозначений $O()$, мы получаем возможность воспользоваться так называемым **асимптотическим анализом**. А в рамках этого подхода можно, например, безоговорочно утверждать, что если n асимптотически приближается к бесконечности, то алгоритм, характеризующийся показателем $O(n)$, проявляет себя лучше по сравнению с алгоритмом $O(n^2)$, тогда как единственное число, полученное с помощью эталонного тестирования, не может служить обоснованием подобного утверждения.

Система обозначений $O()$ позволяет создать абстрактное представление, в котором не учитываются постоянные коэффициенты, благодаря чему она становится более простой в использовании, но менее точной, чем система обозначений $T()$. Например, в конечном итоге алгоритм $O(n^2)$ всегда будет считаться худшим по сравнению с алгоритмом $O(n)$, но если бы эти два алгоритма характеризовались значениями $T(n^2+1)$ и $T(100n+1000)$, то фактически алгоритм $O(n^2)$ был бы лучше при $n \leq 110$.

Несмотря на этот недостаток, асимптотический анализ представляет собой инструментальное средство, наиболее широко используемое для анализа алгоритмов. Этот метод можно считать достаточно точным, поскольку в процессе анализа создается абстрактное представление и для точного количества операций (поскольку игнорируется постоянный коэффициент k), и для точного содержимого входных данных (поскольку рассматривается исключительно их объем n); благодаря этому анализ становится осуществимым с помощью математических методов. Система обозначений $O()$ представляет собой хороший компромисс между точностью и простотой анализа.

Изначально сложные и недетерминированные полиномиальные задачи

С помощью анализа алгоритмов и системы обозначений $O()$ мы получаем возможность рассуждать об эффективности конкретного алгоритма. Однако эти методы не позволяют определить, может ли существовать лучший алгоритм для рассматриваемой задачи. В области **анализа сложности** исследуются задачи, а не алгоритмы. Первая широкая градация в этой области проводится между задачами, которые могут быть решены за время, измеряемое полиномиальным соотношением, и задачами, которые не могут быть решены за время, измеряемое полиномиальным соотношением, независимо от того, какой алгоритм для этого используется. Класс полиномиальных задач (которые могут быть решены за время $O(n^k)$ для некоторого k) обозначается как P . Эти задачи иногда называют “легкими”, поскольку данный класс содержит задачи, имеющие такую продолжительность прогона, как $O(\log n)$ и $O(n)$. Но он содержит и задачи с затратами времени $O(n^{1000})$, поэтому определение “легкий” не следует понимать слишком буквально.

Еще одним важным классом является NP (сокращение от Nondeterministic Polynomial) — класс недетерминированных полиномиальных задач. Задача относится к этому классу, если существует алгоритм, позволяющий выдвинуть гипотезу о возможном решении, а затем проверить правильность этой гипотезы с помощью полиномиальных затрат времени. Идея такого подхода состоит в том, что если бы можно было воспользоваться сколь угодно большим количеством процессоров, чтобы проверить одновременно все гипотезы, или оказаться крайне удачливым и всегда с первого раза находить правильную гипотезу, то NP -трудные задачи стали бы P -трудными задачами. Одним из самых важных нерешенных вопросов в компьютерных науках является то, будет ли класс NP эквивалентным классу P , если нельзя воспользоваться бесконечным количеством процессоров или способностью находить правильную гипотезу с первого раза. Большинство специалистов в области компьютерных наук согласны с тем, что $P \neq NP$, иными словами, что задачи NP являются изначально трудными и для них не существует алгоритмов с полиномиальными затратами времени. Но это утверждение так и не было доказано.

Ученые, пытающиеся найти ответ на вопрос о том, эквивалентны ли классы P и NP , выделили подкласс класса NP , называемый **NP -полными** задачами. В этой формулировке слово “полный” означает “являющийся наиболее ярким представителем” и поэтому указывает на самые трудные задачи из класса NP . Было доказано, что либо все NP -полные задачи принадлежат к классу P , либо ни одна из них к нему не относится. Таким образом, данный класс представляет определенный теоретический интерес, но он важен также с точки зрения практики, поскольку известно, что

многие серьезные задачи являются NP-полными. В качестве примера можно указать задачу установления выполнимости: если дано высказывание пропозициональной логики, то есть ли такой вариант присваивания истинностных значений пропозициональным символам в высказывании, при котором оно становится истинным? Если не произойдет чудо и не совпадут друг с другом классы P и NP, то нельзя будет найти алгоритм, который позволяет решить все задачи установления выполнимости за полиномиальное время. Но исследователей в области искусственного интеллекта в большей степени интересует то, существуют ли алгоритмы, действующие достаточно эффективно при решении типичных задач, выбранных с помощью заранее заданного распределения; как было показано в главе 7, существуют алгоритмы наподобие WalkSAT, которые действуют вполне успешно при решении многих задач.

Класс Σ_1 **ко-NP** является комплементарным (дополнительным) по отношению к классу NP; это означает, что для каждой задачи принятия решения из класса NP существует соответствующая задача в классе ко-NP, на которую может быть дан положительный или отрицательный ответ, противоположный ответу на задачу класса NP. Известно, что P является подмножеством и NP, и ко-NP, кроме того, считается, что к классу ко-NP относятся некоторые задачи, не входящие в класс P. Такие задачи называются Σ_1 **ко-NP-полными** и являются самыми трудными задачами в классе ко-NP.

Класс Σ_1 **#P** (произносится как “шарп P”, или “диз P”) представляет собой множество задач подсчета количества вариантов, соответствующих задачам принятия решения из класса NP. Задачи принятия решения имеют однозначный (положительный или отрицательный) ответ; примером задачи такого типа является следующая: “Существует ли решение для данной формулы 3-SAT?” Задачи подсчета количества вариантов имеют целочисленный ответ, например: “Сколько решений существует для данной формулы 3-SAT?” В некоторых случаях задача подсчета количества вариантов намного труднее по сравнению с соответствующей задачей принятия решения. Например, принятие решения о том, имеет ли двухдольный граф идеальное паросочетание, может быть выполнено за время $O(VE)$ (где V — количество вершин; E — количество ребер графа), тогда как задача подсчета того, какое количество идеальных паросочетаний имеется в этом двухдольном графе, является Σ_1 **#P**-полной. Это означает, что она не менее трудна, чем любая задача из класса Σ_1 **#P**, и поэтому по меньшей мере столь же трудна, как и любая задача NP.

Исследователи определили также класс задач PSPACE; таковыми являются задачи, для которых требуется объем пространства, определяемый полиномиальной зависимостью, даже при их прогоне на недетерминированной машине. Считается, что PSPACE-трудные задачи решаются хуже по сравнению с NP-полными задачами, но не исключено, что в ходе дальнейших исследований может быть установлено, что класс NP эквивалентен классу PSPACE, и также то, что класс P эквивалентен классу NP.

А.2. ВЕКТОРЫ, МАТРИЦЫ И ЛИНЕЙНАЯ АЛГЕБРА

В математике сформулировано определение Σ_1 **вектора** как элемента векторного пространства, но мы будем использовать более конкретное определение: вектор — это упорядоченная последовательность значений. Например, в двухмерном пространстве могут быть определены такие векторы, как $\mathbf{x} = \langle 3, 4 \rangle$ и $\mathbf{y} = \langle 0, 2 \rangle$. В этом

приложении соблюдаются обычные соглашения об обозначении в нем векторов с помощью полужирных символов, хотя некоторые авторы отмечают имена векторов с помощью стрелок или знаков надчеркивания: \vec{x} или \overline{y} . Элементы вектора обозначаются с помощью подстрочных индексов: $\mathbf{z} = \langle z_1, z_2, \dots, z_n \rangle$.

Двумя фундаментальными операциями над векторами являются векторное сложение и скалярное умножение. Векторное сложение $\mathbf{x} + \mathbf{y}$ — это поэлементная сумма: $\mathbf{x} + \mathbf{y} = \langle 3 + 0, 4 + 2 \rangle = \langle 3, 6 \rangle$, а скалярное умножение — это операция умножения каждого элемента на некоторую константу: $5\mathbf{x} = \langle 5 \times 3, 5 \times 4 \rangle = \langle 15, 20 \rangle$.

Длина вектора обозначается как $|\mathbf{x}|$ и вычисляется путем извлечения квадратного корня из суммы квадратов элементов: $|\mathbf{x}| = \sqrt{3^2 + 4^2} = 5$. Точечное произведение (называемое также скалярным произведением) двух векторов, $\mathbf{x} \cdot \mathbf{y}$, представляет собой сумму произведений соответствующих элементов; иными словами,

$$\mathbf{x} \cdot \mathbf{y} = \sum_i x_i y_i,$$

или в данном конкретном случае: $\mathbf{x} \cdot \mathbf{y} = 3 \times 0 + 4 \times 2 = 8$.

Векторы часто интерпретируются как направленные отрезки прямых (напоминающие стрелки) в n -мерном евклидовом пространстве. В таком случае операция сложения векторов эквивалентна совмещению конца одного вектора с началом другого, а точечное произведение $\mathbf{x} \cdot \mathbf{y}$ эквивалентно выражению $|\mathbf{x}| |\mathbf{y}| \cos \theta$, где θ — угол между \mathbf{x} и \mathbf{y} .

✂ **Матрица** — это прямоугольный массив значений, упорядоченных по строкам и столбцам. Ниже показана матрица \mathbf{m} с размерами 3×4 .

$$\begin{pmatrix} m_{1,1} & m_{1,2} & m_{1,3} & m_{1,4} \\ m_{2,1} & m_{2,2} & m_{2,3} & m_{2,4} \\ m_{3,1} & m_{3,2} & m_{3,3} & m_{3,4} \end{pmatrix}$$

Первый подстрочный индекс термина $m_{i,j}$ определяет строку, а второй — столбец. В языках программирования терм $m_{i,j}$ часто записывается как $m[i, j]$ или $m[i][j]$.

Сумма двух матриц определяется путем сложения соответствующих элементов, поэтому $(\mathbf{m} + \mathbf{n})_{i,j} = m_{i,j} + n_{i,j}$. (Если матрицы \mathbf{m} и \mathbf{n} имеют разные размеры, то их сумма не определена.) Можно также определить операцию умножения матрицы на скаляр: $(c\mathbf{m})_{i,j} = cm_{i,j}$. Операция умножения матриц (получения произведения двух матриц) является более сложной. Произведение \mathbf{mn} определено, только если \mathbf{m} имеет размеры $a \times b$, а \mathbf{n} — размеры $b \times c$ (т.е. вторая матрица имеет количество строк, совпадающее с количеством столбцов первой матрицы); результатом является матрица с размерами $a \times c$. Это означает, что операция умножения матриц не коммутативна — в общем случае $\mathbf{mn} \neq \mathbf{nm}$. Если матрицы имеют приемлемые размеры, то результат их умножения являются следующим:

$$(\mathbf{mn})_{i,k} = \sum_j m_{i,j} n_{j,k}$$

Единичная матрица \mathbf{I} имеет элементы $\mathbf{I}_{i,j}$, равные 1, если $i=j$, и равные 0 в противном случае. Она обладает таким свойством, что $\mathbf{mI} = \mathbf{m}$ при всех \mathbf{m} . Транспозиция матрицы \mathbf{m} , которая записывается как \mathbf{m}^T , образуется путем превращения строк в столбцы и наоборот, или, более формально, путем выполнения операции $\mathbf{m}^T_{i,j} = m_{j,i}$.

Матрицы используются для решения систем линейных уравнений с помощью процедуры, называемой алгоритмом ~~за~~ удаления элементов Гаусса—Джордана, который характеризуется показателем $O(n^3)$. Рассмотрим следующее множество уравнений, для которого можно найти решение в терминах x , y и z :

$$\begin{aligned} +2x + y - z &= 8 \\ -3x - y + 2z &= -11 \\ -2x + y + 2z &= -3 \end{aligned}$$

Эту систему уравнений можно представить в виде следующей матрицы:

$$\begin{pmatrix} x & y & z & c \\ 2 & 1 & -1 & 8 \\ -3 & -1 & 2 & -11 \\ -2 & 1 & 2 & -3 \end{pmatrix}$$

Здесь строка $x \ y \ z \ c$ не входит в состав матрицы; она служит лишь в качестве напоминания, к чему относится каждый столбец. Известно, что если обе стороны уравнения будут умножены на константу или если это уравнение будет сложено с другим уравнением, то полученное уравнение останется действительным. Метод удаления элементов Гаусса—Джордана действует по принципу повторного выполнения подобных операций таким образом, чтобы вначале была удалена первая переменная (x) из всех уравнений, кроме первого, а затем эти действия продолжались в форме удаления i -й переменной из всех уравнений, кроме i -го, для всех i . Удалим x из второго уравнения, умножив первое уравнение на $3/2$ и сложив его со вторым. В результате будет получена следующая матрица:

$$\begin{pmatrix} x & y & z & c \\ 2 & 1 & -1 & 8 \\ 0 & .5 & .5 & 1 \\ -2 & 1 & 2 & -3 \end{pmatrix}$$

Продолжая аналогичные действия, удалим x , y и z и получим следующее:

$$\begin{pmatrix} x & y & z & c \\ 1 & 0 & 0 & 2 \\ 0 & 1 & 0 & 3 \\ 0 & 0 & 1 & -1 \end{pmatrix}$$

Таким образом, решением является $x=2$, $y=3$, $z=-1$ (проверьте это решение!).

А.3. РАСПРЕДЕЛЕНИЯ ВЕРОЯТНОСТЕЙ

Вероятность — это мера, заданная на множестве событий, которая удовлетворяет трем приведенным ниже аксиомам.

1. Мера каждого события находится в пределах от 0 до 1. Это утверждение записывается как $0 \leq P(E=e_i) \leq 1$, где E — случайная переменная, представляющая событие; e_i — возможные значения E . Вообще говоря, случайные переменные обозначаются прописными буквами, а их значения — строчными.
2. Мера всего множества равна 1; это означает, что

$$\sum_{i=1}^n P(E=e_i) = 1.$$

3. Вероятность объединения несовместимых событий равна сумме вероятностей отдельных событий; это означает, что $P(E=e_1 \vee E=e_2) = P(E=e_1) + P(E=e_2)$, где e_1 и e_2 являются несовместимыми.

Вероятностная модель состоит из пространства выборок взаимоисключающих возможных результатов, наряду с вероятностной мерой для каждого результата. Например, в модели погоды на завтра результатами могут быть *sunny* (солнце), *cloudy* (облачность), *rainy* (дождь) и *snowy* (снег). Подмножество этих результатов представляет собой событие. Например, событие выпадения осадков — это подмножество $\{rainy, snowy\}$.

Для обозначения вектора значений $\langle P(E=e_1), \dots, P(E=e_n) \rangle$ используется терм $P(E)$. Кроме того, $P(e_i)$ применяется как сокращенное обозначение для $P(E=e_i)$, а $\sum_e P(e)$

служит для обозначения

$$\sum_{i=1}^n P(E=e_i).$$

Условная вероятность $P(B|A)$ определяется как $P(B \cap A) / P(A)$. Переменные A и B являются условно независимыми, если $P(B|A) = P(B)$ (или, равным образом, если $P(A|B) = P(A)$). Непрерывные переменные имеют бесконечное количество значений, и если распределение вероятностей этих значений не характеризуется наличием пиковых значений в отдельных точках, то вероятность любого отдельно взятого значения равна 0. Поэтому определим **функцию плотности вероятностей**, которая также обозначается как $P(X)$, но имеет немного иной смысл по сравнению с дискретной функцией вероятностей $P(A)$. Функция плотности вероятностей $P(X=c)$ определяется как отношение вероятности того, что X попадает в интервал вокруг c , к ширине этого интервала, измеряемая в пределе приближения ширины интервала к нулю:

$$P(X=c) = \lim_{dx \rightarrow 0} P(c \leq X \leq c + dx) / dx$$

Функция плотности должна быть неотрицательной при всех x и соответствовать следующему требованию:

$$\int_{-\infty}^{\infty} P(X) dx = 1$$

Может быть также определена **кумулятивная функция плотности вероятностей** $F(X)$, которая представляет собой вероятность того, что некоторая случайная переменная меньше x :

$$F(X) = \int_{-\infty}^x P(X) dx$$

Следует отметить, что функция плотности вероятностей измеряется в определенных единицах, а дискретная функция вероятностей является безразмерной. Например, если переменная X измеряется в секундах, то плотность измеряется в Гц (т.е. $1/c$). Если \mathbf{x} — точка в трехмерном пространстве, измеряемом в метрах, то плотность измеряется в $1/m^3$.

Одним из наиболее важных распределений вероятностей является **гауссово распределение**, известное также под названием **нормального распределения**. Гауссово распределение со средним μ и среднеквадратичным отклонением σ (и поэтому с дисперсией σ^2) определяется следующей формулой:

$$P(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{(x-\mu)^2}{2\sigma^2}}$$

где x — непрерывная переменная, изменяющаяся в пределах от $-\infty$ до $+\infty$. Если среднее $\mu=0$ и дисперсия $\sigma^2=1$, то имеет место частный случай нормального распределения, называемый **стандартным нормальным распределением**. Если распределение задано на векторе \mathbf{x} в пространстве с d измерениями, то оно представляет собой **многомерное гауссово распределение**:

$$P(\mathbf{x}) = \frac{1}{\sqrt{(2\pi)^n |\Sigma|}} e^{-\frac{1}{2} (\mathbf{x}-\boldsymbol{\mu})^T \Sigma^{-1} (\mathbf{x}-\boldsymbol{\mu})}$$

где $\boldsymbol{\mu}$ — вектор средних; Σ — **матрица ковариации** этого распределения.

При наличии одного измерения можно также определить функцию **кумулятивного распределения** $F(x)$ как вероятность того, что случайная переменная будет меньше чем x . Для стандартного нормального распределения эта функция задается следующим образом:

$$F(x) = \int_{-\infty}^x P(x) dx = \frac{1}{2} \left(1 + \operatorname{erf} \left(\frac{x-\mu}{\sigma\sqrt{2}} \right) \right)$$

где $\operatorname{erf}(x)$ — так называемая **функция ошибок**, не имеющая представления в замкнутой форме.

В **центральной предельной теореме** утверждается, что среднее n случайных переменных приближается к нормальному распределению по мере того, как n стремится к бесконечности. Такому свойству подчиняется почти любая коллекция случайных переменных, при том условии, что значение дисперсии любого конечного подмножества переменных не доминирует над значениями дисперсии других конечных подмножеств.

БИБЛИОГРАФИЧЕСКИЕ И ИСТОРИЧЕСКИЕ ЗАМЕТКИ

Система обозначений $O()$, которая в наши дни широко используется в компьютерных науках, была впервые определена в контексте теории чисел немецким математиком П.Г. Н. Бахманом [57]. Понятие NP-полноты было предложено Куком [289], а современный метод определения способа сведения одной проблемы к другой предложен Карпом [772]. И Кук, и Карп получили за свою работу премию Тьюринга — высшую награду в компьютерных науках.

В число классических работ по анализу и проектированию алгоритмов по праву входят [8] и [809]; более современные достижения отражены в [295] и [1489]. Эти

книги в основном посвящены проектированию и анализу алгоритмов решения разрешимых задач. Сведения в области теории NP-полноты и других форм неразрешимости приведены в книгах Гэри и Джонсона [520], а также Пападимитриу [1168]. В своей книге Гэри и Джонсон не только изложили теоретические основы, но и привели примеры, наглядно показывающие, по каким причинам специалисты в области компьютерных наук считают бесспорным утверждение, что линии водораздела между разрешимыми и неразрешимыми задачами проходит по границе между полиномиальной и экспоненциальной временной сложностью. Кроме того, эти ученые представили объемистый каталог задач, известных как NP-полные или неразрешимые по каким-то другим критериям.

К числу хороших учебников по теории вероятностей относятся [122], [254], [463] и [1310].

Б ОБЩИЕ СВЕДЕНИЯ О ЯЗЫКАХ И АЛГОРИТМАХ, ИСПОЛЬЗУЕМЫХ В КНИГЕ

Б.1. ОПРЕДЕЛЕНИЕ ЯЗЫКОВ С ПОМОЩЬЮ ФОРМЫ БЭКУСА–НАУРА

В настоящей книге определено несколько языков, включая языки пропозициональной логики (с. 1), логики первого порядка (с. 1) и подмножество английского языка (с. 1). Формальный язык определяется как множество строк, в котором каждая строка представляет собой последовательность символов. Все языки, которые были необходимы нам в этой книге, состоят из бесконечного множества строк, поэтому нужен краткий способ, позволяющий охарактеризовать это множество. Для этого служит **грамматика**. Авторы приняли в качестве способа оформления грамматик формальную систему, называемую **формой Бэкуса–Наура (Backus–Naur form — BNF)**. Грамматика BNF состоит из четырех компонентов, перечисленных ниже.

- Множество **терминальных символов**. Это — символы или слова, из которых состоят строки языка. В качестве таких символов могут использоваться буквы (**A, B, C, ...**) или слова (**a, aardvark, abacus, ...**).
- Множество **нетерминальных символов**, которые обозначают компоненты предложений языка. Например, нетерминальный символ *NounPhrase* (именное словосочетание) в английском языке обозначает бесконечное множество строк, которое включает “you” и “the big slobbery dog”.
- **Начальный символ**, который представляет собой нетерминальный символ, обозначающий законченные строки языка. В описанной выше грамматике

английского языка таковым является символ *Sentence*; в грамматике арифметических выражений может быть задан символ *Expr*.

- Множество **правил подстановки** в форме $LHS \rightarrow RHS$, где *LHS* — нетерминальный символ; *RHS* — последовательность, в которую входят от нуля и больше символов (терминальных или нетерминальных).

Правило подстановки в приведенной ниже форме означает, что при наличии двух строк, обозначенных как *NounPhrase* (именное словосочетание) и *VerbPhrase* (глагольное словосочетание), их можно соединить друг с другом и обозначить результат как *Sentence*:

$$Sentence \rightarrow NounPhrase \ VerbPhrase$$

Несколько правил с одинаковой левой частью могут быть заданы как одно правило с той же левой частью и со всеми правыми частями этих правил, разделенными символом $|$. Ниже приведена грамматика BNF для простых арифметических выражений.

$$\begin{aligned} Expr &\rightarrow Expr \ Operator \ Expr \mid (\ Expr \) \mid Number \\ Number &\rightarrow Digit \mid Number \ Digit \\ Digit &\rightarrow 0 \mid 1 \mid 2 \mid 3 \mid 4 \mid 5 \mid 6 \mid 7 \mid 8 \mid 9 \\ Operator &\rightarrow + \mid - \mid \times \end{aligned}$$

Более подробные сведения о языках и грамматиках приведены в главе 22. Следует отметить, что в других книгах в системе обозначений BNF могут использоваться немного другие правила, например, нетерминальные символы обозначаются как $\langle Digit \rangle$ вместо *Digit*, терминальные символы — 'word', а не **word**, а в правилах используются символ $::=$ вместо \rightarrow .

Б.2. ОПИСАНИЕ АЛГОРИТМОВ С ПОМОЩЬЮ ПСЕВДОКОДА

В настоящей книге подробно определено свыше 80 алгоритмов. Авторы решили не останавливаться на каком-то одном языке программирования (и рисковать тем, что читатели, не знакомые с этим языком, не смогут воспользоваться приведенными здесь алгоритмами) и вместо этого представить алгоритмы на псевдокоде. Основные конструкции этого псевдокода должны быть знакомы пользователям таких языков, как Java, C++ и Lisp. В некоторых местах для описания вычислений используются математические формулы или текст на естественном языке, поскольку в противном случае пришлось бы применять более громоздкие конструкции. Необходимо также отметить, что в алгоритмах используются приведенные ниже соглашения.

- **Статические переменные.** Ключевое слово *static* используется для указания на то, что переменной присваивается начальное значение при первом вызове некоторой функции, после чего это значение (или значение, присвоенное переменной с помощью выполненных в дальнейшем операторов присваивания) сохраняется при всех последующих вызовах функции. Таким образом, статические переменные напоминают глобальные переменные в том, что они сохраняют свое значение после каждого вызова содержащей их функции, но остаются доступными только в этой функции. В программах агентов, приведенных в данной книге, статические переменные используются в качестве

“оперативной памяти”. Программы со статическими переменными могут быть реализованы в объектно-ориентированных языках, таких как Java и Smalltalk, в виде “объектов”. В функциональных языках они могут быть реализованы с помощью функциональных замыканий в той среде, в которой определены требуемые переменные.

- **Функции как значения.** Имена функций и процедур начинаются с прописных букв, а имена переменных состоят из строчных букв и выделяются курсивом. Поэтому чаще всего вызов функции выглядит наподобие $F_n(x)$. Однако допускается, чтобы значением переменной была функция; например, если значением переменной f является функция вычисления квадратного корня, то выражение $f(9)$ возвращает 3.
- **Начальный индекс массива, равный 1.** Если не указано иное, то первым индексом массива является 1, как в обычной системе математических обозначений, а не 0, как в языках Java и C.
- **Значимость отступов.** По аналогии с языком Python и в отличие от языков Java и C++ (в которых используются фигурные скобки) или языков Pascal и Visual Basic (в которых используется оператор `end`) для обозначения области действия цикла или условного выражения применяются отступы.

Б.3. ОПЕРАТИВНАЯ ПОМОЩЬ

Большинство алгоритмов, приведенных в данной книге, реализовано на нескольких языках программирования и представлено в нашем оперативном репозитории кода по следующему адресу:

📧 aima.cs.berkeley.edu

Если у вас есть какие-либо комментарии, если вы обнаружили какие-либо неточности или хотите внести предложения по усовершенствованию данной книги, мы всегда будем рады вас выслушать. Посетите указанный выше Web-узел, чтобы ознакомиться с инструкциями по подключению к дискуссионной группе по интересам или отправьте по электронной почте сообщение по следующему адресу:

📧 aima@cs.berkeley.edu

ОПИСАНИЕ СОПРО- ВОЖДАЮЩЕГО WEB-УЗЛА

Авторы книги “Искусственный интеллект. Современный подход. Второе издание” подготовили дополнительные материалы к книге, которые находятся на сопровождающем Web-узле по адресу aima.cs.berkeley.edu. Первая страница этого узла показана на рис. 1.

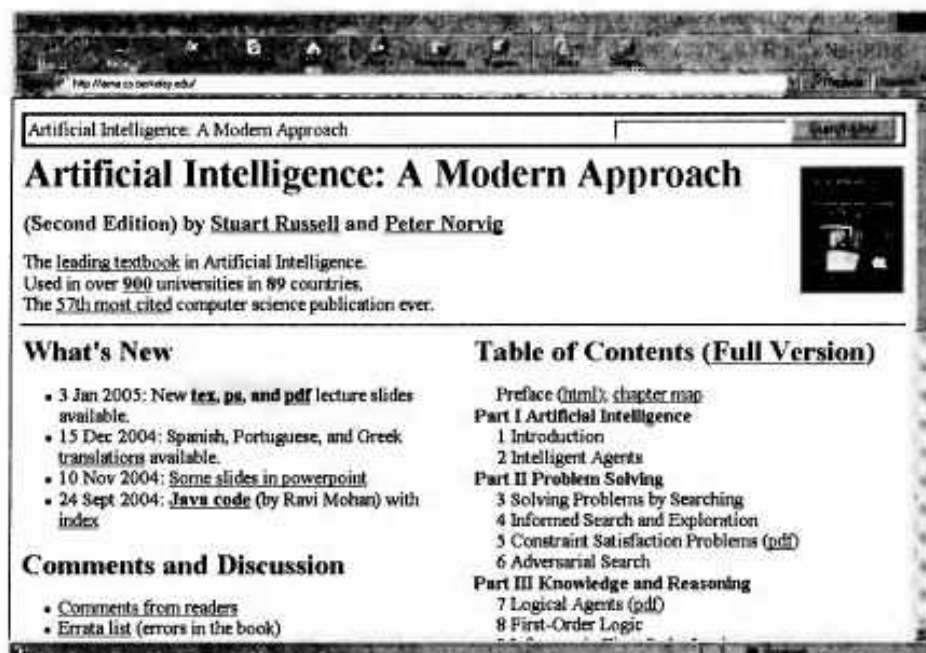


Рис. 1. Первая страница Web-узла aima.cs.berkeley.edu

Материалы, представленные на этом узле, постоянно обновляются и ко времени выхода из печати настоящего издания включают следующее:

- псевдокод алгоритмов, описанных в книге, в формате PDF и PS;
- реализации представленных алгоритмов на нескольких языках программирования (оперативный репозиторий кода);
- данные, предназначенные для проверки программ из репозитория кода;