УДК 681.3.068

В.В. Кручинин

Программные генераторы (обзор)

Рассматривается класс программных систем, называемых генераторами. Описываются основные этапы развития таких систем. Предлагаются обобщенные схемы и направления использования генераторов. Особое внимание уделяется программным генераторам для целей обучения.

Современное программное обеспечение относится к классу самых сложных искусственных систем. Операционные системы, трансляторы, автоматизированные системы управления, проектирования и моделирования — вот неполный перечень таких систем. В настоящее время при описании программной системы, которая синтезирует некоторый информационной объект, стали широко использовать термин «генератор». Ниже предлагается исследовать класс таких программных систем.

Термин «генератор» (лат. *generator* — производитель) возник в технических науках и означает устройство, производящее какой-либо продукт (парогенератор, электрический генератор, генератор импульсов и т.д.) [1]. В теории автоматического управления введено понятие генератора входных возмущений. Это устройство, которое позволяет генерировать по определенному закону входное возмущение для исследования какой-либо системы автоматического регулирования [2]. Рассматривая систему как черный ящик, можно предложить следующую схему (рис. 1). В данной схеме генератор является моделью внешней среды и служит инструментом исследования системы.

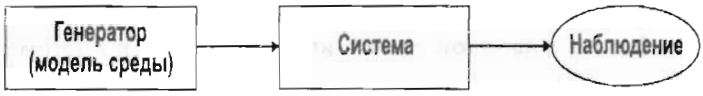

Рис. 1. Исследование модели черного ящика

Развитие систем передачи информации [3, 4], в которых происходит кодирование информации, также приводит к понятию генератора (рис. 2).

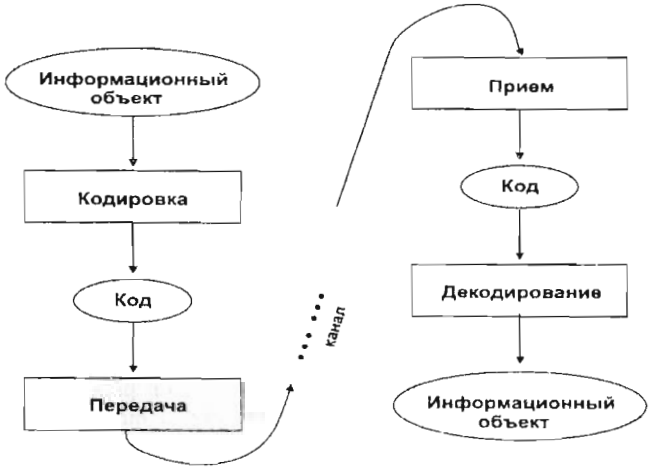

Рис. 2. Генераторы при приеме информации

Первоначально информационный объект кодируется, т.е. ему ставится в соответствие некоторое число (идентификатор). Далее это число передается по каналам связи и поступает на вход декодера. Декодер производит обратную операцию, по заданному коду восстанавливает информационный объект. В некоторых случаях декодер можно рассматривать как генератор, в том смысле, что по некоторому коду (числу, параметру) устройство генерирует информационный объект.

Более сложные схемы использования генераторов возникли при создании технических систем с элементами обучения. На рис. 3 показана обобщенная схема.

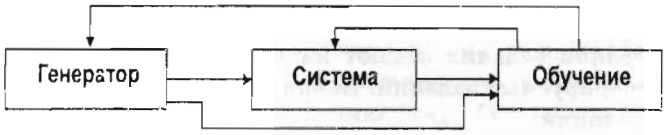

Рис. 3. Схема обучения систем

Генератор на основе конкретных значений входных параметров генерирует входное воздействие и требуемый (желательный) выход системы. Система преобразует входной сигнал в выходной, блок обучения производит сравнение выходных сигналов генератора и системы, устанавливает структуру и значения параметров системы и выдает новые значения для генератора. Процесс обучения завершается на основе некоторого критерия оптимальности.

Примерами таких систем являются корреляционно-экстремальные системы навигации и наведения [5], робототехнические системы [6], нейросистемы и нейрокомпьютеры [7, 8].

Исследования в компьютерных науках, особенно в области моделирования сложных систем, привели к появлению методов Монте-Карло [9], а затем к созданию теории имитационного моделирования [10]. Имитационное моделирование является эффективным инструментом исследования сложных систем. В этом случае строятся упрощенные программные модели, имитирующие как элементы исследуемой системы, так и существенные элементы среды, влияющие на поведение системы. Одним их важнейших элементов систем имитационного моделирования являются датчики случайных чисел (ДСЧ). ДСЧ — это программы, генерирующие псевдослучайные последовательности целых или вещественных чисел. Для исследования различного класса систем появились разнообразные программные генераторы, основанные на использовании ДСЧ. Примерами могут служить генераторы изображений с заданными статистическими свойствами [11], генераторы карт [12], генераторы сценариев [13].

Другими важными классами программных генераторов являются генераторы отчетов [14], широко применяемых в базах данных. Однако данный класс программ нельзя считать в чистом виде генератором, поскольку результат их работы не подается на вход системы, а служит конечным продуктом. Генераторы программ — еще один класс подобных систем, который по заданному описанию на выходе получает программу на некотором языке программирования, например, генератор различного рода трансляторов и интерпретаторов по описанию входного языка [15]. На сайте *www.rvb.ru* дан список программ-генераторов текста различного назначения: генерации русскоязычных *стихоподобных* текстов («инструмент поэта»); генератор письменных жалоб Скотта Пейкина; программы генерации любовных писем; генератор *псевдофилософских* текстов; генератор случайных текстов на основе заданной грамматики для английского языка.

Еще одна сфера применения программных генераторов является борьба с плагиатом в сети интернет [16]. Особенно это касается дистанционного обучения. Подтверждением тому служат цифры, показывающие, что огромное число студентов пользуется многочисленными сайтами, распространяющими рефераты. Имеются сайты, которые представляют услуги по решению задач, курсовых проектов и пр., что приводит к созданию банка ответов и решений на контрольные задания и курсовые работы.

Для решения проблем плагиата некоторые фирмы предлагают программные системы, которые могут анализировать текст на предмет заимствования [17]. Можно также использовать поисковые сервера, вставив в поле окна поиска требуемый фрагмент. Однако это требует существенных затрат на организацию соответствующей проверки.

Опыт, накопленный в Томском государственном университете систем управления и радиоэлектроники (ТУСУР) при организации дистанционного обучения, показывает, что наиболее действенным является выдача уникального, индивидуального задания каждому студенту. Это достигается за счет применения программных генераторов тестовых заданий [18]. Рассмотрим подробнее обобщенную структуру такого генератора, показанную на рис. 4.

База знаний содержит полную информацию для генерации задания. Если задание формулируется в форме задачи, то база знаний содержит банк шаблонов формулировок задач,

области изменения параметров задач, алгоритм нахождения решения или проверки наличия решения, алгоритм изменения значений параметров. Если задание — это последовательность вопросов, то могут использоваться разнообразные модели и алгоритмы генерации вопросов [19].

Параметры генерации задания задают начальные значения параметров генерации и ограничения на тип генерируемых заданий. Начальные значения параметров генерации могут быть получены на основании:

1) датчика случайных чисел;

2) индивидуальных параметров студента: фамилия, имя, отчество, место и дата рождения, семестр и пр.;

3) некоторого унифицированного кода, присваиваемого студенту при поступлении в вуз.

Уникальное тестовое задание является конечной целью работы генератора и представляется на некотором декларативном языке для последующего этапа визуализации. Например, генераторы в системе «Фея» формируют тестовое задание на языке представления теста этой системы[20]. Можно также представлять тестовые задания на языке Tex [21] или XML [22].

База параметров выданных тестовых заданий предназначается для реализации двух основных целей.

Идентифицировать задание, выполненное студентом. Опыт Томского межвузовского центра дистанционного образования показывает, что некоторые студенты присылают ответы или решения не на те задания, которые им были выданы.

Генераторы — сложные программы, отладка которых является достаточно трудоемким делом. Проверить работоспособность его во всех случаях практически невозможно. Поэтому желательно иметь такую базу для оперативного анализа и исправления ошибок.

Алгоритм работы *генератора* зависит от типа задания и моделей представления знаний. Обобщенная схема работы показана на рис. 5.

Рис. 4. Обобщенная структура генератора заданий

Рис. 5. Алгоритм работы генератора заданий

Рассмотрим основные шаги работы алгоритма.

1. Выбор параметров и элементов задания — это, собственно говоря, шаг, на котором производится синтез задания. Выбор значений параметров и элементов задания осуществляется, как правило, на основе использования датчика случайных чисел. Это в некоторых случаях существенно снижает параметры генерирующего алгоритма. Если параметры генерировать от входных параметров студента, то необходимо подбирать хеш-функции, которые бы вычисляли значения ключей. Хорошие результаты дает использование алгоритмов нумерации, которые перечисляют все варианты заданий данного алгоритма генерации [23].

2. Проверка семантики задания — это важный шаг, поскольку не всякая комбинация параметров может привести к решению задачи. В простейшем случае необходимо проверить некоторую формулу, например, подкоренное выражение не должно быть меньше нуля. В более сложных случаях необходимо разрабатывать специальный алгоритм семантического контроля.

3. Формулировка задания — шаг, на котором производится преобразование внутреннего представления тестового задания на некоторый декларативный язык. Поскольку текст задания может содержать формулы, рисунки, таблицы и пр., то нужно использовать язык описания документа для какого-либо конкретного пакета.

Развитие идей искусственного интеллекта, воплощение их в реальных системах привело к появлению более сложных схем. На рис. 6 представлена обобщенная схема, состоящая из генератора, планировщика, решателя и советчика [24]. *Генератор* обеспечивает построение некоторого обучающего воздействия или учебного задания, которое поступает на вход системы. Генерация воздействия или задания осуществляется на основе параметров, которые устанавливает *Планировщик*. Последний выполняет построение заданной последовательности учебных заданий, после выполнения которых можно сказать, что система обучена.

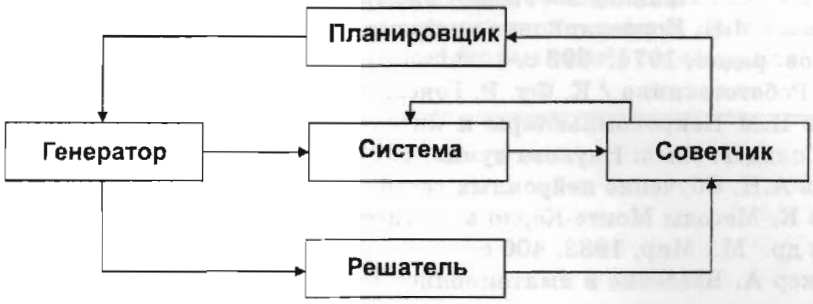

Рис. 6. Схема обучения интеллектуальных систем

Полученное задание поступает на вход *Системы* и *Решателя*. *Решатель* обеспечивает стандартное выполнение задания, состоящее из некоторой последовательности элементарных актов. Важно отметить, что эта последовательность не единственна, а может быть некоторое множество последовательностей.

Результаты выполнения задания системой и решателя поступают на вход *Советчика*. Он обеспечивает сравнение результатов работы системы и решателя, устраняет неточности и ошибки в работе системы при выполнении задания. После того, как система обеспечивает заданное качество выполнения задания, *Советчик* выдает сообщение *Планировщику* для продолжения обучения. Завершение обучения осуществляет *Планировщик* в соответствии с прохождением плана обучения и достижения заданного критерия обучения.

Здесь за кадром осталась база знаний, существенный элемент интеллектуальной системы. Понятно, что ни генератор, ни планировщик, ни советчик, ни решатель не могут обойтись без базы знаний. Предполагается, что каждый из указанных модулей обращается к своей или интегрированной базе знаний.

Еще одна область применения генераторов — стимулирование творческой (созидательной, креативной) деятельности человека [25, 26]. Создаются специальные программные системы, так называемые генераторы идей, которые обеспечивают стимулирование творческой деятельности человека. Например, программное обеспечение фирмы *Creax* обеспечивает анализ и решение проблем, определение выгод и убытков от принятия решений в сфере бизнеса и менеджмента.

В заключение, рассмотрев основные схемы использования генераторов, можно выделить следующее:

1. Генератор является важной частью схемы исследования систем, частным случаем таких исследований является тестирование, которое может иметь различные цели, например, для технических систем — оценивание технических параметров, для программных систем — обнаружение ошибок или доказательство их отсутствия. Для человека — это определение его психофизического состояния или определение уровня знаний.

2. Генератор является важной частью схемы обучения систем. Он позволяет генерировать множество учебных примеров. Обучение систем также можно классифицировать на обучение технических, программных и организационных систем.

3. Генератор может использоваться в системах кодирования и декодирования информации. Здесь генераторы позволяют решать задачи декодирования при сжатии, шифровании и повышении надежности передачи информации.

4. Генератор позволяет решить проблемы плагиата и шпаргалок при организации дистанционного обучения, получая уникальные индивидуальные задания для каждого студента.

5. Генератор идей используется для усиления творческой деятельности человека (решение проблем, изобретательство и др.). Существенно расширяет границы поиска решений.

Литература

1. Словарь иностранных слов. М.: Рус. яз., 1989. 662 с.

2. Первозванский А.А. Курс теории автоматического управления: Учеб. пособие. М.: Наука, 1986. 615 с.

3. Игнатов В.А. Теория информации и передачи сигналов. М.: Радио и связь, 1991. 279 с.

4. Дмитриев В.И. Прикладная теория информации. М.: Высшая школа, 1989. 319 с.

5. Белоглазов И.Н. Корреляционно-экстремальные системы / И.Н. Белоглазов, В.П. Тарасенко. М.: Сов. радио, 1974. 392 с.

6. Фу К. Робототехника / К. Фу, Р. Гонсалес, К. Ли. М.: Мир, 1989. 621 с.

7. Амосов Н.М Нейрокомпьютеры и интеллектуальные роботы / Н.М. Амосов, Т.Н. Байрон, А.Д. Гольцев. Киев: Наукова думка, 1991. 271 с.

8. Горбань А.Н. Обучение нейронных сетей. М.: С.П. «ПараГраф», 1990. 56 с.

9. Биндер К. Методы Монте-Карло в статистической физике / К. Биндер, Д. Сиперли, Ж.-П. Ансен и др. М.: Мир, 1982. 400 с.

10. Прицкер А. Введение в имитационное моделирование и язык СЛАМ II. М.: Мир, 1997. 646 с.

11. Буймов А.Г. Корреляционно-экстремальная обработка изображений. Томск: Изд-во Том. гос. ун-та, 1987. 132 с.

12. Цветков В.Я. Геоинформационные системы и технологии. Сер. «Диалог с компьютером». М.: Финансы и статистика, 1998. 286 с.

13. Кнут Д. Искусство программирования для ЭВМ. Т. 2. Получисленные алгоритмы / Ред. пер. К.И. Бабенко: Пер. Г.П. Бабенко, Э.Г. Белага, Л. В. Майоров. М.: Мир, 1977. 723 с.

14. Баас Р. Delphi 4: полное руководство / Р. Баас, М. Фервай, Х. Гюнтер: Пер. с нем. Киев: BHV, 1998. 800 с.

15. Маккиман У. Генератор компиляторов: Пер. с англ./ У. Маккиман, Дж. Хорнинг, Д. Уортман. М.: Статистика,1980. 527 с.

16. Степанов С.А. Преподавание политических наук в Канаде // Вестник Российского университета дружбы народов. Сер.: Политология. 2003. № 4. С. 118–124.

17 Groark M. Term paper Mills, Anti-Plagiarism Tools, and Academic Integrity / M. Groark, D. Oblinger, M. Choa // EDUCOUSE Review September–October 2001. Pp. 40–48.

18. Исакова О.Ю. Опыт организации контроля знаний в Томском межвузовском центре дистанционного образования / О.Ю. Исакова, В.В. Кручинин, А.Ф. Уваров // Тез. докл. междунар. науч.-метод. конф. «Инновационные технологии организации обучения в техническом вузе: на пути к новому качеству образования». Пенза: Изд-во ПГУАС, 2004. С. 220–222.

19. Башмаков А.И. Разработка компьютерных и обучающих систем / А.И. Башмаков, И.А. Башмаков. М.: Информационно-издательский дом «Филинъ», 2003. 616 с.

20. Кнут Д. Компьютерная типография. М.: Мир, 2003. 668 с.

21. Маршал Б. XML в действии: Практика программирования: Пер. с англ. / Б. Маршал. М.: ТРИУМФ, 2002. 366 с.

22. Дмитриева Н.М. Технология разработки компьютерных учебных работ и экзаменаторов в Томском межвузовском центре дистанционного образования / Н.М. Дмитриева, В.В. Кручинин, Е.А. Ситникова // Тез. докл. семинара «Электронные учебники и учебно-методические разработки в открытом образовании». М.: Изд-во МЭСИ, 2000. С. 69–70.

23. Кручинин В.В. Генераторы в компьютерных учебных программах. Томск: Изд-во Том. гос. ун-та, 2003. 200 с.

24. Кручинин В.В. Разработка компьютерных учебных программ. Томск: Изд-во Том. гос. ун-та, 1998. 211 с.

25. Альтшуллер Г. С. Найти идею: Введение в теорию решения изобретательских задач / Генрих Саулович Альтшуллер; Ред. А.К. Дюнин; АН СССР СО. 2-е изд., доп. Новосибирск: Наука: СО, 1991. 225 с.

26. Половинкин А.И. Автоматизация поискового конструирования (искусственный интеллект в машинном проектировании) / А.И. Половинкин, Н.К. Бобков, Г.Я. Буш / Под ред. А.И. Половинкина. М.: Радио и связь, 1981. 344 с.

Кручинин Владимир Викторович
Канд. техн. наук, зав. лабораторией инструментальных систем моделирования и обучения ТУСУРа
Телефон (служебный): (3822) 41 36 74 доп. 170
Эл. почта: kru@ie.tusur.ru

V.V. Kruchinin

Programm generators (summary)

Program systems class, also called generators, is considered within the article. General development phases are described as applied for such systems. Generalized schemes are proposed as well as possible application directions. Key point of the document is featured to program generators, which are developed for particularly for educational needs.