Математическая логика и логическое программирование

mk.cs.msu.ru → Лекционные курсы
→ Математическая логика и логическое программирование (3-й поток)

Блок 52

Формальная верификация

Лектор:
Подымов Владислав Васильевич
E-mail:
valdus@yandex.ru

ВМК МГУ, 2022/2023, осенний семестр

Вступление

В математической логике есть ещё много такого, что было бы полезно и интересно обсудить, но не получится, так как время уже на исходе

Поэтому остановимся подробно только на одном разделе: на том, как можно при помощи логических методов решить очень нетривиальную и очень полезную прикладную задачу, *казалось бы* не связанную с логикой

Обсу́дим в деталях одну такую задачу из области программирования

Ошибки в программах

«**Любая нетривиальная программа содержит хотя бы одну ошибку**» (*автор неизвестен*)

Правильная программа[1] не содержит ошибок и решает задачу, которую требовалось решить при помощи этой программы

Эти утверждения выглядят разумно, но порождают ряд нетривиальных вопросов, например:

- Что такое «ошибка», насколько плохо её наличие, и нужно ли её вообще искать?
- Можно ли как-нибудь убедиться, что ошибок в программе нет и она действительно решает поставленную задачу?

1 Из утверждения выше следует, что такой программы не существует среди нетривиальных. Но можно и пофантазировать

Ошибки в программах

Подход к проверке правильности работы программы,
про который знает каждый программист — это **тестирование**:

- придумывается показательный набор входных данных программы (тестовое покрытие)
- программа выполняется на тестовом покрытии
- результаты выполнения программы сравниваются с ответами к задаче, которые считаются правильными

Основной недостаток такого подхода:
ошибки всё равно остаются в программе, хотя их и становится меньше

Кроме того, в современном мире широко применяются вычислительные системы, для которых даже после «качественного» тестирования
не гарантируется отсутствие критичных ошибок:
аппаратные и программно-аппаратные, интерактивные и распределённые системы, сложные программные комплексы, ...

Ошибки в программах

В некоторых случаях протестировать код — это приемлемо («*более-менее работает, а дальше пусть пользователь разбирается*»), но далеко не всегда: даже от небольших ошибок серьёзно зависит прибыль компаний[1], успешность проектов мирового масштаба[2], быт людей[3] и даже их жизни[4]

---

1 1994. Процессор Intel, ошибка в реализации деления чисел с плавающей точкой ⇒ замена дефектных процессоров, сотни миллионов $ убытка

2 1962–сейчас. Ракеты и спутники, взорвавшиеся и исчезнувшие из-за программных ошибок: Фобос (пропущена кавычка), Ariane 5 (ошибка в округлении чисел), Mars Global Surveyor (перепутаны английская и метрическая системы мер), Hitomi (ошибка в программе стабилизации вращения), ...

3 2003. Полное отключение электричества в нескольких областях США и Канады ⇐ ошибка в реализации взаимодействия программ оповещения об электрической нагрузке (race condition)

4 1980-е гг. Пять смертей при лечении рака аппаратом Therac-25 ⇐ ошибочное увеличение мощности радиационного облучения при очень редком сочетании времён выполнения параллельных подпрограмм (race condition)

Формальная верификация

Есть и дугой подход к проверке правильности вычислительных систем:

- формулируется набор требований к выполнению системы, означающих «система функционирует правильно»
- **строго** доказывается или опровергается утверждение о том, что система удовлетворяют этим требованиям

Требования такого вида, записанные на математическом языке, называются формальной спецификацей системы, и соответствующий язык — языком спецификаций

Проверка соблюдения требований с помощью математических методов называется формальной верификацией системы

Если в качестве языка спецификаций выбрать какой-либо **логический язык**, то для проверки правильности программы можно будет использовать **логические методы**