



**ПОЛИТЕХ**

Санкт-Петербургский  
политехнический университет  
Петра Великого



**ИКНТ**

# Верификация и анализ программ Bounded Model Checking

2020

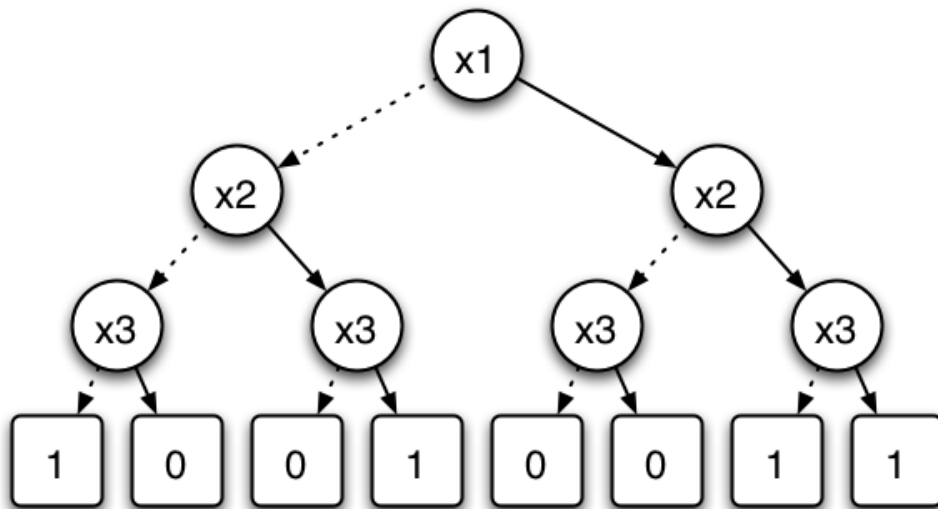


КОМПЬЮТЕРНЫЕ СИСТЕМЫ И ПРОГРАММНЫЕ ТЕХНОЛОГИИ

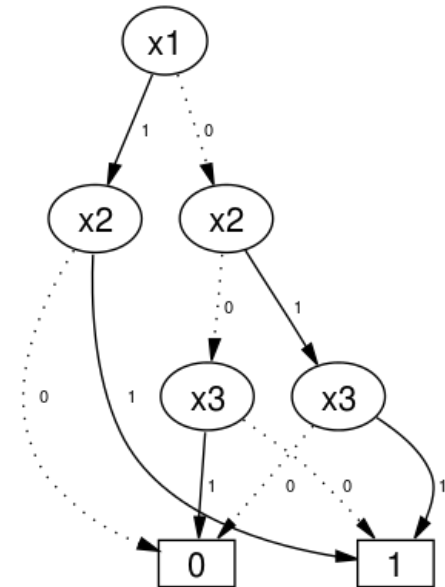
# Bounded Model Checking

- ▶ Bounded Model Checking (BMC) – ограниченная проверка модели
- ▶ Метод, изобретенный в 1999 году
- ▶ “Bounded” – ограниченный, подразумевает ограничение пространства состояний
- ▶ Наиболее эффективные алгоритмы классического Model Checking:
  - Используют символьную верификацию
  - Используют BDD для представления логических формул

# Бинарные решающие диаграммы (BDD)



x1	x2	x3	f
0	0	0	1
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	0
1	1	0	1
1	1	1	1

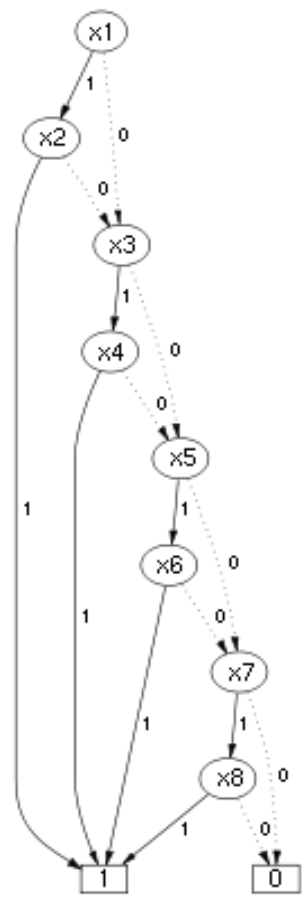
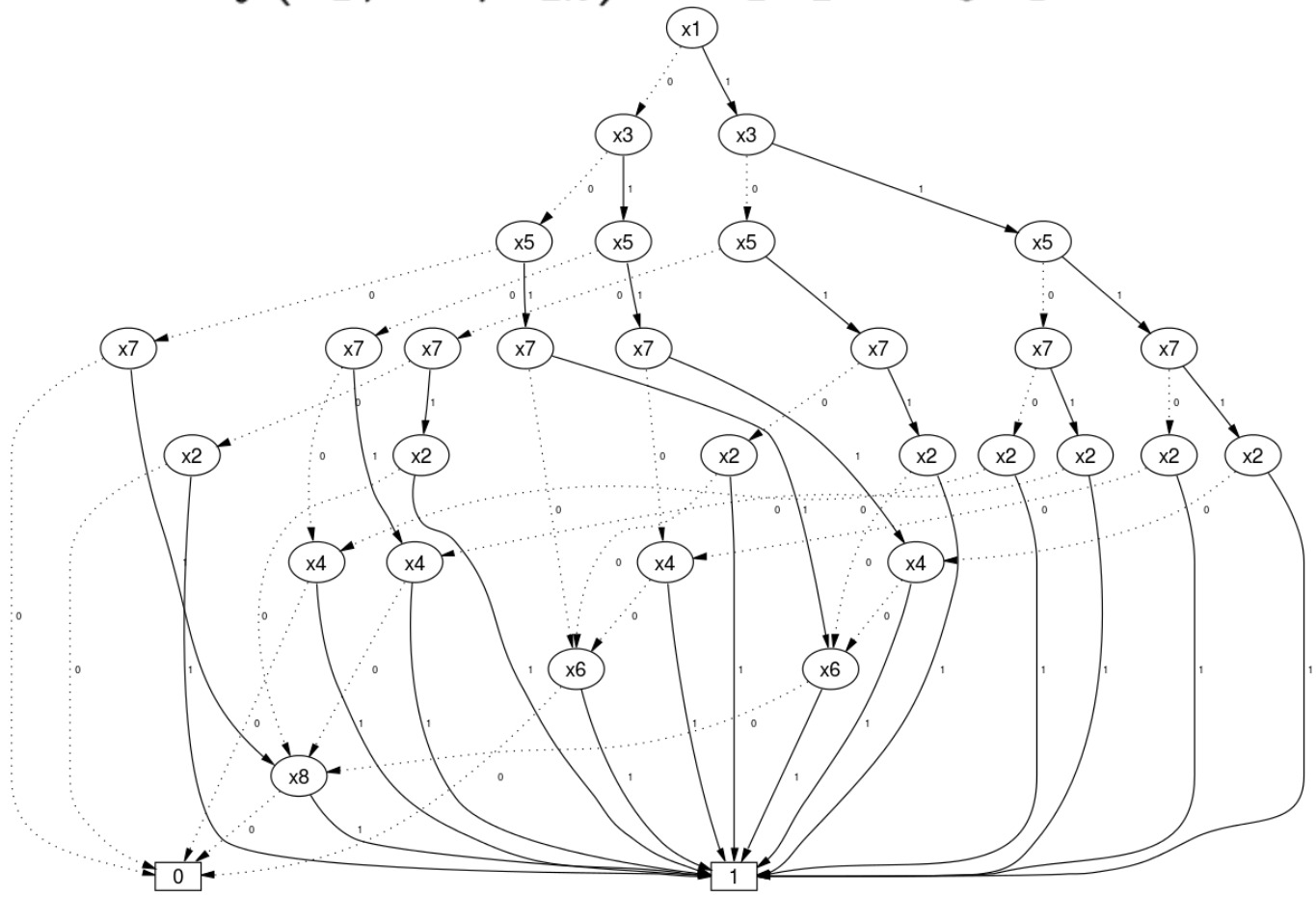


# Проблемы Model Checking

- ▶ Эффективность очень зависит от “правильного” порядка переменных в BDD
- ▶ Алгоритм выбора “правильного” порядка переменных - NP-полный
- ▶ Часто требуется вмешательство человека для определения порядка переменных
- ▶ Размер пространства состояний часто неприемлем для современных средств

# Примеры BDD с разным порядком переменных

$$f(x_1, \dots, x_{2n}) = x_1x_2 + x_3x_4 + \dots + x_{2n-1}x_{2n}$$



# Bounded Model Checking

- ▶ Основные идеи:
  - Сократить пространство состояний
  - Просматривать ограниченное число шагов
  - Использовать вместо BDD и символьной верификации методы решения задачи SAT-выполнимости
- ▶ Не страдает проблемой взрыва пространства состояний
- ▶ Существуют очень эффективные реализации

# Особенности классического ВМС

- ▶ ВМС использует процедуру SAT вместо BDD
  1. Фиксируется глубина (граница)  $k$ , например  $k=1$
  2. Формируется конечное (глубиной  $k$ ) дерево решений
  3. Строится формула, которая выполняется тогда и только тогда, когда имеется контрпример длины  $k$
  4. Если контрпример найден – алгоритм завершается
  5. Если контрпример не найден, то  $k := k+1$
  6. Граница достигнута? Нет - переход к поиску более длинного контрпримера (шаг 2)
  7. После определенного числа итераций можно убедиться, что контрпримера не существует и спецификация выполняется

# Особенности классического ВМС

- ▶ Поиск доказательства формулы  $E_f$  сводится к поиску контрпримера для формулы  $A \neg f$  и наоборот
- ▶ Благодаря особенностям алгоритма поиска контрпример находится быстро
- ▶ ВМС находит контрпример **минимальной длины**
  - удобно для отладки и быстрого поиска ошибок
  - после нахождения контрпримера нет смысла анализировать дальше



# Использование ВМС для поиска ошибок в программах

- ▶ Проверяемая функция программы переводится в CFG в форме SSA
- ▶ Все циклы раскручиваются на predetermined глубину путем подстановки (loop unrolling)
- ▶ Получившаяся модель без циклов превращается в логическую формулу:
  - Присваивание переменным в SSA встречается один раз
  - Каждый узел SSA заменяется логической формулой
  - В итоге вся модель функции превращается в гигантскую формулу P
- ▶ Проверяемые обобщенные свойства программы кодируются предикатами (**C**) в точках проверки:
  - Выход за границу массива, корректность указателя и т.п.
  - Истинность предиката показывает нарушение правила корректности программы

# Использование ВМС для поиска ошибок в программах

- ▶ Результирующая формула  $F = P \wedge C$  проверяется на разрешимость с помощью SMT-солвера
- ▶ Если  $F$  – разрешима (sat), то программа некорректна, а вектор решений – контрпример
- ▶ Если  $F$  – неразрешима (unsat), то операция повторяется для модели с увеличенной глубиной раскрутки циклов и рекурсий, до тех пор, пока не будет достигнут максимум. В итоге, если осталось unsat – программа корректна, относительно свойства  $C$

# Использование ВМС для поиска ошибок в программах

- ▶ ВМС позволяет дополнительно поверять:
  - Выполнение контрактов
  - Выполнение assertion
- ▶ Для контрактов:
  - Пред- и постусловия записываются в форме предикатов логики первого порядка
  - Проверяется разрешимость формулы  $f'(Pred, P, Post)$
  - Если формула разрешима – контракт не выполняется
- ▶ Для assertion – аналогично

# Основные сложности ВМС при анализе программ

- ▶ Представление сложных структур данных
- ▶ Потенциальная незавершаемость запуска SMT-решателя
- ▶ Межпроцедурность
- ▶ Вопросы определения глубины раскрутки циклов и рекурсий

# Использование ВМС для поиска ошибок в программах

- ▶ Анализаторы, основанные на ВМС:
  - LLBMC (Karlsruhe Institute of Technology)
  - CBMC (Oxford)
  - ESBMC (Университет Саутгемптона, Университет Штелленбош и Федеральный Университет Амазонии)
  - SMT-ВМС
  - Borealis (СПбПУ)