**Міністерство освіти і науки України**

**Запорізький національний технічний університет**

кафедра програмних засобів

**РЕФЕРАТ**

з дисципліни «Проектування інформаційних систем» на тему:

**«МЕТОДИ ТА ІННСТРУМЕНТИ ПЕРЕВІРКИ МОДЕЛІ»**

Виконала:

студентка групи КНТ-715 А.Ю. Фролова

Прийняв :

к.т.н, доцент Г.В. Табунщик

**Зміст**

[1 Введення 3](#_Toc526935264)

[2 Глава 1. Перевірка моделей 5](#_Toc526935265)

[2.1 Перевірка моделей 5](#_Toc526935266)

[2.2 Процес верифікації моделей програм 5](#_Toc526935267)

[2.3 Модель Крипке 6](#_Toc526935268)

[2.4 Темпоральні логіки 7](#_Toc526935269)

[2.5 Деякі інструментальні засоби 8](#_Toc526935270)

[3 Список літературы 9](#_Toc526935271)

# Введення

Широко поширена думка про те, що «допомогти комп'ютерам принести нам ще більше користі» і перекласти на них все більш складні і тонкі завдання важко не з-за недостатньої швидкодії і незадовільних обчислювальних можливостей існуючих машин, а з-за нашої обмеженої здатності проектувати та створювати складні системи з досить високим ступенем впевненості в їх коректності при будь-яких обставин.

Завдання перевірки правильності проектованої системи, тобто її забезпечення її коректності починаючи з самих ранніх етапів проектування, є основною проблемою в процесі розробки будь-якої надійної системи, і зусилля, витрачені на її рішення, поглинають все зростаючу частку вартості і проектувального циклу.

В даний час у багатьох фірмах для перевірки правильності проекту практикуються методи імітаційного моделювання та тестування. Хоча вони зарекомендували себе досить ефективними на дуже ранніх стадіях налагодження, коли проектована система все ще наповнена численними помилками, результативність цих методів швидко знижується, як тільки проектована система стає чистіше. При цьому зростаючий час, який потрібно для виявлення все більш тонких помилок, призводить в сум'яття. Застосування зазначених методів суттєво ускладняється тим, що ніхто не в силах дізнатися, коли їх можливості вже вичерпані, або хоча б оцінити, скільки помилок може все ще бути в проектованої системі. Оскільки складність проектованих пристроїв зростає надмірно, скажімо, від 5 тисяч до 5 мільйонів вентилів у кристалі, деякі далекоглядні керівники проектів починають усвідомлювати, що традиційні методи не дають ніякої можливості збільшити масштаби проектів і можуть призвести до повного провалу.

Вельми привабливою і все більш привабливою альтернативою імітаційному моделюванню і тестуванню стають методи формальної верифікації. Оскільки при імітаційному моделюванні та тестуванні задіяні лише деякі з можливих сценаріїв поведінки проектованої системи, залишається відкритим питання про те, чи не міститься фатальної помилки в недосліджених траєкторіях; а ось формальна верифікація забезпечує вичерпний аналіз всіх можливих варіантів поведінки системи. Таким чином, якщо проектована система, піддана формальної верифікації, була визнана коректною, то це означає, що всі варіанти її поведінки були розглянуті, і запитань про те, наскільки повно було проведено дослідження і не пропущені які-небудь особливості поведінки, просто не виникає.

У порівнянні з іншими підходами метод перевірки на моделі володіє двома чудовими перевагами:

\* Він повністю автоматизований, і його застосування не вимагає від користувача ніяких особливих знань в таких математичних дисциплінах, як

логіка і теорія доказа теорем. Всякий, хто може провести моделювання проектованої системи, цілком здатний здійснити і перевірку цієї системи. У рядку сформованих практичних технологій метод перевірки на моделі може розглядатися як, безумовно, найкращий засіб імітаційного моделювання.

• Якщо проектована система не володіє бажаними властивостями, то в процесі перевірки на моделі завжди буде побудований контрприклад, який демонструє поведінку системи, що спростовує цю властивість. Ця помилкова траса надає безцінну інформацію для розуміння дійсної причини помилки, так само як і важливий ключ до вирішення виниклої проблеми.

Ці дві чудових гідності, а також відкриття символьних методів верифікації моделей, що дозволяють проводити вичерпний неявний перебір в просторі з астрономічно великим числом станів, здійснили революційний переворот в області формальної верифікації і перевели її з суто академічної дисципліни в дієздатну практичну технологію, яка потенційно може бути впроваджена в багато промислові процеси розробки в якості цінного додаткового методу для перевірки правильності проектованих систем.

Достатнім свідченням того, що у промисловості широко визнаний величезний практичний потенціал методу перевірки на моделі, служить велика кількість дослідників і розробників, які створюють оригінальні засоби перевірки на моделі і займаються їх застосуванням у більшості великих компаній по виробництву сучасних напівпровідникових пристроїв і процесорів.

# Глава 1. Перевірка моделей

## Перевірка моделей

Перевірка моделей — метод автоматичної формальної верифікації паралельних систем з кінцевим числом станів. Одна з переваг, пов'язаних з вказаним обмеженням, полягає в тому, що перевірка може бути здійснена чисто автоматично. Зазвичай процедура верифікації

полягає у вичерпному обході простору станів системи, для того щоб з'ясувати, чи виконується специфікація. При наявності достатніх ресурсів ця процедура завжди завершується з відповіддю "так" або "ні". Більш того, вона може бути реалізована досить ефективним алгоритмом, здатним працювати на обчислювальних машинах невисокого класу (але зазвичай все-таки не на стандартних персональних комп'ютерах).

Хоча обмеження, пов'язане з кінцевим числом станів системи, виглядає вельми істотним недоліком, метод верифікації моделей застосуємо до багатьох важливих класів обчислювальних систем. До таких відносяться контролери, а також багато-комунікаційні протоколи. У деяких випадках системи з нескінченним числом станів можуть бути перевірені цим методом у поєднанні з різноманітними правилами абстракції та індукції. Нарешті, нерідко помилки можна виявити, обмежуючи структури даних до спеціальних випадків, коли число станів звичайно. Наприклад, програма з необмеженою чергою повідомлень може бути налагоджена, якщо обмежитися розглядом черг невеликого розміру (два або три).

## Процес верифікації моделей програм

Застосування методу перевірки на моделі складається з декількох етапів:

1) Моделювання

Перше завдання полягає в приведенні проектованої системи до такого формального виду, який був би прийнятний для інструментальних засобів верифікації моделей програм. У багатьох випадках це просто завдання компіляції. В інших випадках, при обмеженнях за часом та обсягом пам'яті, моделювання може зажадати абстракції, щоб позбутися від несуттєвих деталей, що не відносяться до справи.

2) Специфікація

Перед проведенням верифікації потрібно сформулювати властивості, якими повинна володіти проектована системи. Зазвичай специфікації задаються на мові формальної логіки. Для апаратури і програмного забезпечення, як правило, застосовують темпоральну логіку, що дозволяє описувати, як поведінка системи протікає в часі.

Важливим питанням специфікації є повнота. Метод перевірки на моделі дає можливість переконатися, що модель проектованої системи відповідає заданої специфікації, проте визначити, охоплює задана специфікація всі властивості, якими має володіти система, неможливо.

3) Верифікація

В ідеальному випадку верифікація проводиться повністю автоматично. Однак на практиці вона часто вимагає сприяння людини. Однією зі сторін діяльності людини є аналіз результатів верифікації. Білі результати перевірки негативні, то користувачеві нерідко надають трасу, що містить помилку. Вона будується в якості контрпримера для перевіряється властивості і може допомогти проектувальнику простежити, де виникає помилка. В цьому випадку аналіз помилкової траси може спричинити за собою модифікацію системи і повторне застосування алгоритму перевірки на моделі.

## Модель Крипке

В якості моделі зазвичай використовується, так звана, модель Крипке. Це недетермінований кінцевий автомат застосовуваний при перевірці моделей для представлення поведінки системи. Модель представляється орієнтованим графом вершини якого описують досяжні стану системи, а ребра - переходи з стану в стан. Функція позначок зіставляє кожній вершині безліч властивостей які виконуються у відповідному стані. Більш формально, моделлю Крипке над безліччю елементарних висловлювань P називається система переходівS = (S, s0, →,L), де:

* S – кінцева множина станів;
* s0 ∈ S – начальний стан;
* → ⊆ S × S – тотальне відношення переходів (тотальність означає, що для кожного стану s ∈ S має існувати стан s 0 ∈ S, для якого має місце (s, s 0 ) ∈ →, s → s);
* L : S → 2 P – функція, що позначає кожен стан безліччю елементарних висловлювань, справжніх в цьому стані.

Шлях у структурі Крипке зі стану s0 – це нескінченна послідовність станів π = s0s1s2 . . . така, що для всіх i ≥ 0 виконується si → si+1. Для деякого шляху π = s0s1s2s3 . . . позначимо π I суфікс π, який виходить видаленням з π перших I станів – наприклад, π 1 = s1s2s3 . . ., а π(i) буде позначати i-е стан шляху, π(0) = s0, π(1) = s1 і т. д.

## Темпоральні логіки

Спочатку темпоральні логіки були розроблені філософами для дослідження того, як феномен часу проявляється в міркуваннях природною мовою. Темпоральні логіки

є формалізмом, який призначений для опису послідовностей переходів між станами реагуючої системи. Вони відіграють важливу роль у формальній верифікації. Темпоральні логіки дозволяють виражати властивості поведінки різних програмних і апаратних систем. Такими властивостями, наприклад, є властивості «процес ніколи не досягне тупикового стану», в кожному можливому стані програми виконується умова p» або «в будь-якому випадку програма обов'язково досягне стану, в якому буде виконуватися умова p».

Темпоральні логіки зазвичай класифікуються відповідно до того, чи є структура часу лінійної або ветвящейся. Одними з найбільш популярних логік для специфікації та верифікації властивостей програмних систем є логіка CTL (branching time-logic або computation tree logic) і логіка лінійного часу LTL (linear time-logic). Для цілей верифікації автоматних програм логіка LTL заслуговує особливої уваги, оскільки будь-яка формула в рамках даної логіки являє собою автомат Бюхі, що приймає нескінченні допустимі шляхи структури Кріпке, яка в свою чергу задає поведінку (всі можливі виконання) перевіряється на коректність «автоматною» програми. Що дозволяє при специфікації і верифікації "автоматних" програм оперувати в основному таким простим поняттям, як «автомат».

Різниця між логіками LTL і CTL проявляється в тому, як вони відносяться до розгалуження в дереві обчислень досліджуваної програми. У логіці ветвящегося часу темпоральні оператори знаходяться безпосередньо під дією кванторов по тих шляхах, які виходять із заданого стану. У логіці лінійного часу оператори призначені для опису подій протягом єдиного шляху обчислення.

## Деякі інструментальні засоби

Для проведення перевірки на моделі необхідні програмні засоби, що дозволяють автоматично провести перевірку. В якості таких засобів доцільно використовувати вже існуючі пакети

прикладних програм-верифікаторів, які розробляються і підтримуються провідними науковими лабораторіями і центрами на протязі досить тривалого часу (більше десяти років). Серед програмних засобів верифікації класичним методом перевірки моделі можна виділити, наприклад, SPIN, SMV, CADP і CPN Tools (який буде детально розглянуто в наступному розділі). При цьому кожен верифікатор має свій формалізм для завдання моделі і свій спосіб породження структури Крипке для цієї моделі. Більш того, верифікатори мають і свою модифікацію (реалізацію) темпоральних логіки, яка (у ряді випадків) може виявитися менш виразною, ніж, наприклад, темпоральні логіки CTL і LTL.

# Список літературы

1. Кларк Э.М., Грамберг О., Пелед Д. Верификация моделей программ: Model Checking. – М.: МЦНМО, 2002.
2. Модель Крипке [Електронний ресурс] – Режим доступу до ресурсу: http://www.cruer.com/max7612/Модель\_Крипке.