Міністерство освіти і науки України

Харківський національний університет радіоелектроніки

Факультет __Комп'ютерної інженерії та управління_____
(повна назва)

Кафедра Автоматизації проектування обчислювальної техніки
(повна назва)

КВАЛІФІКАЦІЙНА РОБОТА
Пояснювальна записка

рівень вищої освіти _____другий (магістерський)_____
(рівень вищої освіти)

АВТОМАТ ВЕКТОРНО-ДЕДУКТИВНОГО МОДЕЛЮВАННЯ НЕСПРАВНОСТЕЙ ЛОГІКИ
(тема)

Виконав: студент II курсу, групи СКСм-22-2
_____Кулак Г.К. _____
(прізвище, ініціали)

Спеціальність
_____*123 – Комп'ютерна інженерія*_____
(код і повна назва спеціальності)

Тип програми
_____*освітньо-професійна*_____
(освітньо-професійна або освітньо-наукова)

Освітня програма
Спеціалізовані комп'ютерні системи
(повна назва освітньої програми)

Керівник проф. Чумаченко С.В.

Допускається до захисту

Зав. каф. АПОТ  Чумаченко С.В.
(підпис) (прізвище, ініціали)
2023 р.

Харківський національний університет радіоелектроніки

Факультет Комп'ютерної інженерії та управління

Кафедра Автоматизації проектування обчислювальної техніки

Рівень вищої освіти другий (магістерський)

Спеціальність 123_ Комп'ютерна інженерія
(шифр і назва)

Тип програми Освітньо-професійна
(освітньо-професійна або освітньо-наукова)

Освітня програма Спеціалізовані комп'ютерні системи
(повна назва

ЗАТВЕРДЖУЮ:
Зав. кафедри АПОТ

_____Чумаченко С.В.
(підпис)
«____»_____ 2023 р.

ЗАВДАННЯ
НА КВАЛІФАКАЦІЙНУ РОБОТУ

студентові Кулаку Георгію Костянтиновичу
(прізвище, ім'я, по батькові)

1. Тема роботи Автомат векторно-дедуктивного моделювання несправностей логіки
затверджена наказом по університету від 03__ листопада_2023 р. № 1282 Ст
2. Термін подання студентом роботи до екзаменаційної комісії _19_ _січня__ 2024 р.
3. Вихідні дані до роботи
Основні поняття, визначення та метрики комп'ютингу
Методи моделювання несправностей
Елементи булевої логіки

4. Перелік питань, що потрібно опрацювати у роботі
Аналіз предметної області
Дедуктивні моделі та методи
Автомат векторно-дедуктивного моделювання несправностей логіки

5. Перелік графічного матеріалу із зазначенням креслеників, схем, плакатів, комп'ютерних ілюстрацій (слайдів)
слайди презентації – 26

6. Консультанти розділів роботи (п.6 включається до завдання за наявності консультантів згідно з наказом, зазначеним у п.1)

Найменування розділу	Консультант (посада, прізвище, ім'я, по батькові)	Позначка консультанта про виконання розділу	
		підпис	дата

КАЛЕНДАРНИЙ ПЛАН

№	Назва етапів роботи	Термін виконання етапів	Примітка
1	Отримання завдання	01.09.2023-05.09.2023	
2	Аналіз предметної області, сучасних технологічних інновацій	07.09.2023-21.09.2023	
3	Основні поняття, визначення та метрики комп'ютингу	22.09.2023-15.10.2023	
4	Моделювання несправностей на основі таблиць істинності	16.10.2023-06.11.2023	
5	Дедуктивний автомат для моделювання несправностей як адрес	07.11.2023-07.12.2023	
6	Оформлення пояснювальної записки	08.12.2023-30.12.2023	
7	Оформлення графічного матеріалу	02.01.2024-05.01.2024	
8	Перевірка виконаного проекту керівником	06.01.2024-12.01.2024	

Дата видачі завдання 01 вересня 2023 р.

Студент Кулак Г.К.
(підпис)

Керівник роботи (підпис) проф. Хаханов В.І.
(посада, прізвище, ініціали)

РЕФЕРАТ

Пояснювальна записка: 61 с., 31 рис., 4 табл., 51 джерел.

МОДЕЛЬ, МЕТОД, МЕТРИКА, АРХІТЕКТУРА, КОМП'ЮТИНГ, ЦИФРОВА СХЕМА, ДЕДУКТИВНЕ МОДЕЛЮВАННЯ, АВТОМАТ, ВЕКТОРНА ФОРМА ЛОГІКИ, МАТРИЦЯ, ТАБЛИЦЯ, ВЕКТОРНА МОДЕЛЬ ДЕФЕКТІВ І ФУНКЦІЙ.

У магістерській роботі розглядаються питання, пов'язані зі створенням моделей, методів, архітектур, спрямованих на зниження часу верифікації цифрових схем на основі векторного паралельного моделювання несправностей.

Мета дослідження – суттєве зниження часу верифікації цифрових схем за рахунок векторного паралельного моделювання несправностей як адрес.

Для досягнення поставленої мети необхідно вирішити такі задачі:

– проаналізувати сучасні технологічні тенденції;

– виконати аналітичний огляд методів моделювання несправностей;

– вдосконалити автомат векторно-дедуктивного моделювання несправностей логіки.

Об'єкт дослідження – технології синтезу, аналізу, цифрових логічних структур для моделювання несправностей.

Предмет дослідження – моделі, методи і алгоритми синтезу та аналізу цифрових логічних структур.

ABSTRACT

The explanatory note contains: 61 pages, 31 figures, 4 tables, 51 sources according to the list of links.

MODEL, METHOD, METRICS, ARCHITECTURE, COMPUTING, DIGITAL SCHEME, DEDUCTIVE MODELING, AUTOMATA, VECTOR FORM OF LOGIC, MATRIX, TABLE, VECTOR MODEL OF DEFECTS AND FUNCTIONS.

The master's work examines issues related to the creation of models, methods, architectures aimed at significantly reducing the time of verification of digital circuits based on vector parallel modeling of faults.

The purpose of the study is to significantly reduce the time of verification of digital circuits due to vector parallel modeling of faults as addresses.

To achieve the goal, the following tasks must be solved: analyze modern technological trends; perform an analytical review of fault modeling methods; to improve the automaton of vector-deductive modeling of logic malfunctions.

The object of research is the technology of synthesis, analysis, and digital logical structures for fault modeling.

The subject of research is models, methods and algorithms for the synthesis and analysis of digital logical structures.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, ОДИНИЦЬ, СКОРОЧЕНЬ І ТЕРМІНІВ

AI – штучний інтелект (Artificial Intelligent);

AI-ML – штучний інтелект – машинне навчання;

AI TRiSM – ШІ управління довірою, ризиками та безпекою;

API – апаратно-програмного забезпечення;

HDL – Hardware Description Languge;

LEO – низька навколоземна орбіта;

MIPS RISC – конвеєр з обмеженою системою команд;

ML – Machine Learning;

RTL – Register Transaction Level;

STEM – Steps in Threat Exposure Managment;

IaaS – платформи та інфраструктури як послуги;

PaaS (Platform as a Service) – платформа як послуга;

ЕШІ – емоційний штучний інтелект;

ДНФ – диз’юнктивна нормальна форма;

НДДКР – науково-дослідні та дослідно-конструкторські роботи;

ТІ – таблиця істинності;

ЦОД – центр обробки даних;

ШІ – штучний інтелект;

ВСТУП

Розглядаються питання, пов'язані зі створенням моделей, методів, архітектур, спрямованих на суттєве зниження часу верифікації цифрових схем на основі векторного паралельного моделювання несправностей.

Мета дослідження – суттєве зниження часу верифікації цифрових схем за рахунок векторного паралельного моделювання несправностей як адрес – досягається розв’язанням наступних задач:

– аналізуються сучасні технологічні тенденції;

– виконується аналітичний огляд методів моделювання несправностей;

– вдосконалюється автомат векторно-дедуктивного моделювання несправностей логіки.

Розглядається новий метод моделювання одиночних константних несправностей (вхідних, внутрішніх та вихідних) ліній схеми на основі логічних векторів функціональних елементів, які використовуються для побудови дедуктивних векторів транспортування вхідних несправностей на виходи схеми. Метод орієнтований на імплементацію в будь-яку пам'ять на основі виконання read-write транзакцій, що робить його вільним від потужної системи команд центрального процесора та економічним за витратами енергії та часу моделювання несправностей. Несправності розглядаються як адреси для вибору відповідних біт дедуктивних векторів, що дає можливість підвищувати паралелізм обробки векторів несправностей схеми за рахунок збільшення складності логічних елементів. Для обробки схеми із глобальними зворотними зв'язками необхідно вводити псевдозмінні та додаткові цикли обробки вхідних двійкових векторів для приведення результатів моделювання до стабільного вигляду. Для обробки цифрових автоматів необхідно виконати попередній синтез моделі автомата з метою приведення його до векторної форми із псевдозмінними.

1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ

Аналізуються тенденції сучасних електронних технологій на основі огляду літературних джерел та прогнозів американської дослідницької та консалтингової компанії Gartner, що спеціалізується на ринках інформаційних технологій, відома регулярними дослідницькими звітами у форматах «магічний квадрант» та «цикл хайпа».

1.1 Стан питання

Кіберсоціальний комп'ютинг в останні роки займає провідні позиції на ринку human-free менеджменту та електронних технологій, де основними проблемами є технології передбачення соціальних процесів і їх цифрового хмарного оптимального управління на основі точного моніторингу з метою збереження життя та екології планети [1-12].

Gartner Hype Cycle [7] (рис. 1.1) надає уявлення про більш ніж 2000 технологій, які перетворюються у короткий набір обов'язкових для вивчення нових методологій і тенденцій. Акцент Hype Cycle традиційно спрямований на тенденції, які з'являються в першій половині циклу.

Рисунок 1.1 – Gartner Hype Cycle нових технологій

У 2020 році Gartner переорієнтував Hype Cycle на впровадження нових технологій, які раніше не висвітлювалися у пресі. Сьогодні компанії виявляють страхове шахрайство, використовуючи комбінації аналізу претензій, комп'ютерних програм і приватних слідчих. За оцінками ФБР, загальна вартість страхового шахрайства, не пов'язаного з охороною здоров'я, становить близько 40 мільярдів доларів на рік. Розвивається нова технологія, яка називається емоційним штучним інтелектом (ЕШІ), що забезпечує «надлюдські можливості» і дозволяє виявляти страхове шахрайство на основі аудіо аналізу того, хто телефонує. Крім виявлення випадків шахрайства, ЕШІ технологія може поліпшити якість обслуговування клієнтів шляхом відстеження позитивних емоцій, більше точного напрямку побажань абонентів, забезпечення кращої діагностики неспроможності, виявлення водіїв, що відволікаються, і адаптації навчання до поточного емоційного стану студента. Емоційний штучний інтелект, хоча і залишається відносно новим, але є однією з 21 нових технологій, доданих в Gartner Hype Cycle for Emerging Technologies.

У 2019-2020 роках нові технології підрозділялися на п'ять основних тенденцій: зондування (точний моніторинг) і мобільність, розширені можливості людини, постклассичні обчислення (квантові) і зв'язок, цифрові екосистеми і просунутий (емоційний) ШІ і аналітика на його основі.

1. *Сенсорика і мобільність*. Ця тенденція містить технології зі зростаючою мобільністю і здатністю маніпулювати об'єктами навколо них, включаючи 3D-камери і більш досконале автономне водіння. З розвитком датчиків і ШІ автономні роботи будуть краще розуміти навколишній світ. Наприклад, безпілотні літальні апарати для доставки вантажів (літаючі і колісні), зможуть краще орієнтуватися в ситуаціях і маніпулювати об'єктами. Ця технологія поки що обмежена правилами, але її функціональність продовжує розвиватися.

2. *Доповнена людина*. Розширені людські можливості покращують як когнітивні, так і фізичні здібності людського тіла, включаючи біочіпи і ЕШІ.

Деякі з них матимуть «надлюдські здібності», наприклад, протезом, який перевершує силу людської руки, в той час як інші створять роботизовану шкіру, чутливу до дотиків. Ці технології також в кінцевому підсумку забезпечать досвід, який поліпшує здоров'я, інтелект і силу людей. Інші технології в цій тенденції включають в себе: персоніфікацію, розширений інтелект, імерсивні виробничі приміщення та відповідні біотехнології (штучні тканини).

3. *Постклассичні обчислення і зв'язок*. Класичні або виконавчі обчислення, що використовують двійкові біти, розвивалися шляхом внесення змін в існуючі традиційні архітектури. Ці зміни привели до більш швидких процесорів, більш щільною пам'яті і збільшення пропускної здатності. Постклассичні обчислення і комунікації використовують абсолютно нові архітектури, а також поступові поліпшення старих. Тут фігурують 5G, стандарти стільникового зв'язку наступного покоління, які мають нову архітектуру, поділ ядра і бездротовий зв'язок. Ці досягнення дозволяють супутникам на низькій навколоземній орбіті (LEO) працювати на набагато менших висотах, близько 1200 миль або менше, ніж традиційні геостаціонарні системи на відстані близько 22000 миль. Результатом є глобальні широкосмугові або вузькосмугові послуги передачі голосу і даних, в тому числі в районах з невеликим або відсутнім наземним або супутниковим покриттям. Технології цієї тенденції містять пам'ять нового покоління і нанорозмірних 3D-друк.

4. *Цифрові екосистеми* являють собою мережеві з'єднання між суб'єктами, підприємствами, людьми і речами, спільно використовують цифрову платформу. Ці екосистеми розвивалися у міру оцифрування та трансформації традиційних виробничо-побутових треків, забезпечуючи більш плавні і динамічні зв'язки з різними агентами і організаціями в регіонах і галузях. В майбутньому вони стануть децентралізованими автономними організаціями, які працюють незалежно від людей і покладаються на розумні контракти. Цифрові екосистеми постійно розвиваються і об'єднуються, що

призводить до появи нових продуктів і можливостей. Інші технології в цій тенденції включають: DigitalOps, графи знань, синтетичні дані і децентралізовану мережу.

5. *Високий рівень ШІ і аналітика.* Розширена аналітика розглядається як автономна або напівавтономна перевірка даних або контенту з використанням складних інструментів, що виходять за межі традиційних бізнес-уявлень. Формуються нові класи алгоритмів і наука про дані, які ведуть до нових можливостей, до навчання з перенесенням, які використовують раніше навчені моделі машинного навчання як передових відправних точок для нової технології. Розширена аналітика дає можливість глибокого розуміння процесів і явищ, для формування прогнозів і рекомендацій. Інші технології в цій тенденції включають: адаптивне машинне навчання, термінальний штучний інтелект, термінальний інтелектуальний аналіз, зрозумілий штучний інтелект, штучний інтелект PaaS, генеративні змагальні мережі і графічну аналітику.

1.2 Топ-10 головних стратегічних IT-тенденцій 2024 за версією Gartner

Експерти дослідницької та консалтингової компанії Gartner представили дослідження, яке описує десяток головних стратегічних технологічних трендів у 2024 році [2]. Дотримання цих трендів дозволить підвищити стійкість бізнесу, максимізувати цінність даних, залучити та утримати таланти, досягти цілей сталого розвитку, стимулювати зростання та прискорення цифрового бізнесу.

Карта технологічних трендів подана на рис. 1.2. У Gartner представили карту, яка показує, що кожна з тенденцій пов'язана з однією або декількома ключовими темами бізнесу: захист та збереження минулих і майбутніх інвестицій, прийняття правильних рішень у потрібний час та забезпечення цінності для середовища, що змінюється як внутрішніх, так і зовнішніх споживачів.

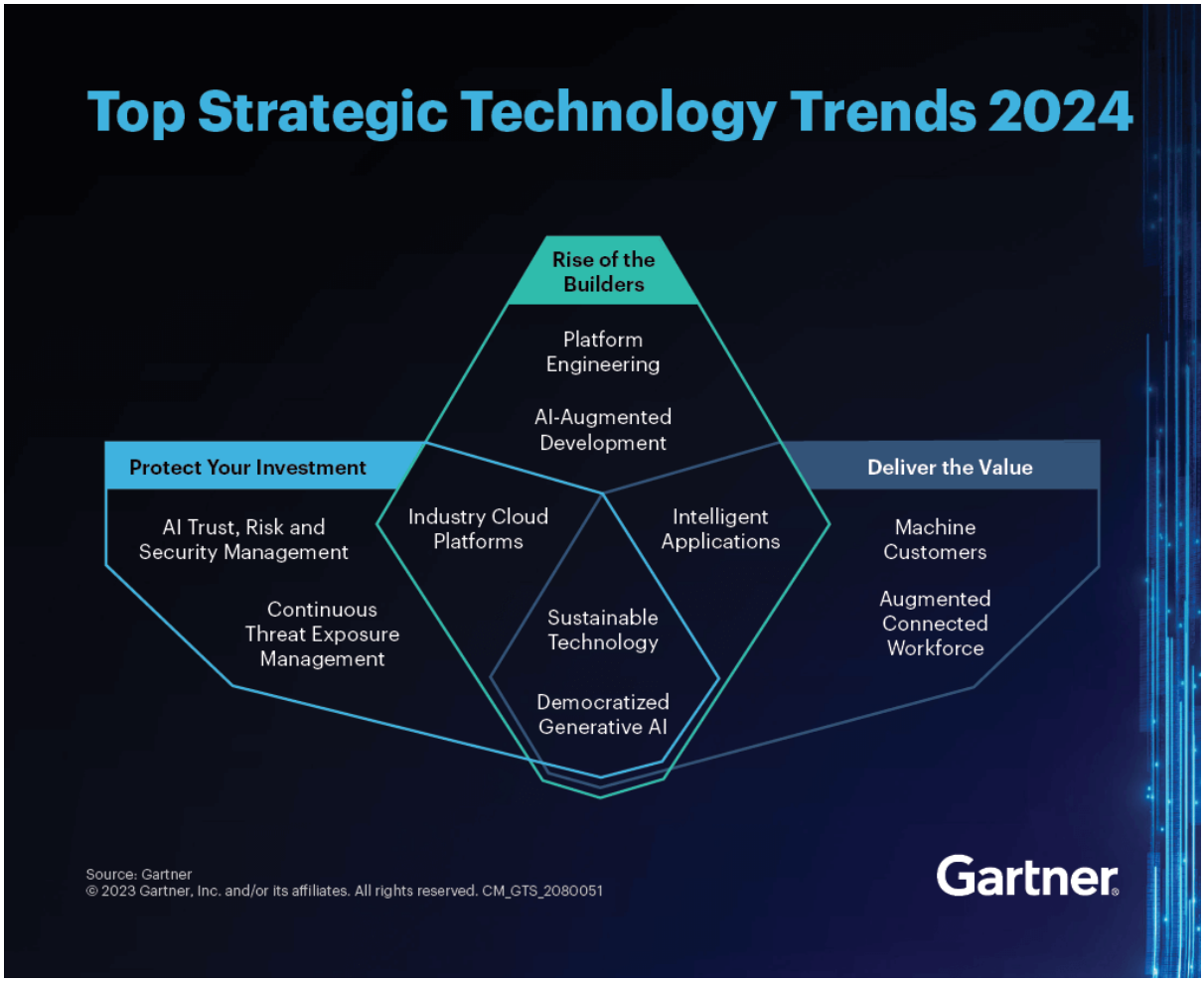


Рисунок 1.2 – Карта технологічних трендів 2024

Карта включає 10 трендів: управління довірою, ризиками та безпекою ШІ (AI TRiSM), безперервне управління ризиками (STEM – Steps in Threat Exposure Managment), стійкі технології, розробка платформ, розробка з використанням штучного інтелекту, галузеві хмарні платформи, інтелектуальні програми, демократизація генеративного штучного інтелекту, розширена підключена робоча сила та клієнти-машини. Розглянемо їх докладніше.

Захист інвестицій. Аналітики Gartner зазначають, що стійкий ефект від інвестицій у технології забезпечується цілеспрямованими зусиллями, розрахунком окупності та далекоглядністю при їх впровадженні. До цієї категорії потрапляють такі IT-тенденції: управління довірою, ризиками та

безпекою ШІ (AI TRiSM); безперервне управління ризиками (STEM); галузеві хмарні платформи; стійкі технології; демократизація генеративного ШІ.

Управління довірою, ризиками та безпекою ШІ. Генеративний ШІ викликав широкий інтерес до пілотних проектів штучного інтелекту, але організації часто не враховують ризики доти, доки моделі або додатки ШІ не почнуть використовуватися на практиці. Комплексна програма управління довірою, ризиками та безпекою ШІ дозволяє заздалегідь інтегрувати управління та забезпечити відповідність, справедливість, надійність систем ШІ та захист конфіденційності даних.

TRiSM для управління ризиками пропонує інструменти зрозумілості та моніторингу роботи моделі, інструменти гнучкого управління ModelOps, а також безпеки та конфіденційності ШІ.

У Gartner прогнозують, що до 2026 року моделі ШІ організацій, які впроваджують TRiSM, досягнуть 50% покращення з погляду бізнес-цілей та визнання користувачами.

Безперервне управління ризиками. Програма безперервного управління ризиками самостійно виявляє та визначає пріоритетність кіберзагроз для бізнесу. Вона дозволяє підприємствам постійно і послідовно оцінювати доступність його фізичних та цифрових активів для хакерів, а також їх схильність до атак.

Покрокове впровадження програми STEM. У Gartner вважають, що до 2026 року організації, які віддають пріоритет інвестиціям у технології у сфері безпеки на основі програм безперервного управління ризиками, втричі рідше зазнають злому.

Галузеві хмарні платформи. Галузеві хмарні платформи призначені для задоволення конкретних потреб тих сегментів галузі, які недостатньо обслуговуються універсальними рішеннями. Такі платформи об'єднують можливості програмного забезпечення, а також платформи та інфраструктури як послуги (IaaS). Вони засновані на загальнодоступних хмарних сервісах, але

пропонують гравцям галузі більш гнучкий спосіб керування робочими навантаженнями.

Під час опитування Gartner у 2022 році серед підприємств Північної Америки та Європи близько 40% респондентів заявили, що вже розпочали впровадження галузевих хмарних платформ, а ще 15% тестують їх у пілотному режимі.

Аналітики очікують, що до 2027 року підприємства використовуватимуть галузеві хмарні платформи для прискорення понад 50% своїх найважливіших бізнес-ініціатив. У майбутньому вони трансформуються в екосистемні хмари, де підприємства зможуть вести бізнес-процеси, такі як закупівля, розподіл, обробка платежів і, можливо, навіть презентувати НДДКР та інновації.

Розумні програми. Інтелектуальні програми пропонують новий досвід клієнтам, користувачам, власникам продуктів та розробникам. Їх впровадження забезпечує зв'язок між безперервною аналітикою та прийняттям рішень, а також дає широкі можливості шляхом впровадження чат-ботів та розумних інтерфейсів. Все це підвищує автоматизацію та забезпечує динамічну трансформацію бізнесу.

ШІ-додатки в бізнесі. Аналітики Gartner прогнозують, що до 2026 року 30% нових програмних додатків будуть використовувати ШІ. Звичайні страхові компанії впроваджуватимуть ШІ з метою оцінки ризиків пошкодження майна, а автостраховики – для оцінки поведінки водіїв.

Стійкі технології. Моделі штучного інтелекту навчаються з використанням енергоємних серверів у центрах обробки даних (ЦОД), які збільшують викиди вуглекислого газу. На ЦОД, де навчають ШІ, вже припадає близько 2% всього споживання електроенергії у США. Компромісом може стати поєднання «ШІ для сталого розвитку» із «стійкістю ШІ». Аналітики вважають, що для його впровадження потрібно вжити таких заходів:

впроваджувати композитний ШІ, який працює на кшталт людського мозку та використовує графи знань, причинно-наслідкові мережі та інші

«символічні» уявлення для більш ефективного вирішення широкого спектру бізнес-завдань;

контролювати споживання енергії під час машинного навчання та припиняти його, коли витрати перестануть виправдовуватися;

зберігати дані для навчання моделей локально, щоб знизити споживання електроенергії та підвищити конфіденційність даних; повторно використовувати вже навчені моделі та впроваджувати більш енергоефективне обладнання;

регулювати робочі навантаження ШІ залежно від регіону, часу доби, погодних умов та інших факторів;

використовувати в ЦОД екологічно чисту енергію та моделювати вплив на навколишнє середовище, а також переваги для бізнесу при розробці стратегії ШІ.

За прогнозами аналітиків, до 2027 року 80% IT-директорів орієнтуватимуться на показники продуктивності, прив'язані до стійкості їхньої організації.

Промисловість 4.0. GitOps та постквантова криптографія: крива хайпа Gartner у 2023 році.

Розкриття цінності спільнот. Аналітики Gartner вважають, що в 2024 році бізнесу потрібно звернутися до творчих здібностей багатьох спільнот, які створюють програми та інші рішення. Кроками цього стануть впровадження технологій галузі, які відповідають конкретним потребам організації та фахівців, розробка «дорожньої карти» для нефахівців та тісна співпраця із зацікавленими сторонами бізнесу для постачання програмного забезпечення. Цьому відповідають такі IT-тенденції: розробка платформ; розробка із використанням технологій штучного інтелекту; галузеві хмарні платформи; інтелектуальні додатки; стійкі технології; демократизація генеративного ШІ.

Розробка платформ. Проектування платформ покращує досвід розробників та підвищує продуктивність, оскільки вони можуть самостійно працювати з автоматизованими процесами, а їхні потреби обслуговує

спеціальна команда інженерів. При цьому кінцевий користувач швидше отримує готовий продукт, а розробка та використання обходяться дешевше. У Gartner очікують, що до 2026 80% організацій, що займаються програмним забезпеченням, створять платформні команди.

Розробка з використанням ШІ. Аналітики зазначають, що інженери-програмісти можуть використовувати ШІ для критично важливих видів діяльності протягом життєвого циклу розробки програмного забезпечення – від планування до тестування. ШІ вже здатний писати код, перекладати його на різні мови програмування, модернізувати додатки та розраховувати витрати на них, розробляти дизайн додатків та багато іншого.

Можливості ШІ у розробці. На думку аналітиків, вже до 2026 року генеративний ШІ суттєво змінить 70% процесів проектування та розробки нових додатків.

Забезпечення цінності. Аналітики зазначають, що адаптація бізнесу до вимог клієнтів, що змінюються, дозволить поліпшити якість обслуговування зацікавлених сторін і збільшити доходи. Цьому сприятимуть підходи, засновані на алгоритмах, а також доступ до цифрових інструментів, що швидко розвиваються. Їм відповідають такі тренди: клієнти-машини; розширена підключена робоча сила; інтелектуальні програми; стійкі технології; демократизація генеративного ШІ.

Клієнти-машини. Клієнти-машини – це пристрої зі штучним інтелектом, які потенційно можуть виступати у ролі споживачів послуг, наприклад, виконуючи функції голосового помічника (замовляючи їжу тощо). У зв'язку зі зростанням ринку таких пристроїв аналітики Gartner радять компаніям робити інформацію про продукти та послуги доступною для ШІ-клієнтів на будь-якому етапі покупки. Цьому сприятимуть:

відкритий доступ до API (апаратно-програмного забезпечення), а також використання інструментів боротьби з ботами;

використання всіх цифрових точок взаємодії, включаючи соціальні мережі, мобільні програми та чат-боти, а також створення клієнтської платформи для спрощеної взаємодії та оплати;

розвиток партнерства між відділами продажів, маркетингу, ланцюжка поставок, ІТ та аналітики, що дозволить розглядати різні сценарії купівельної поведінки та робити ланцюжок поставок досить гнучким, щоб реагувати на несподівані тенденції попиту;

навчання персоналу відділу продажу та обслуговування роботі з агентами ШІ. Співробітники повинні розуміти алгоритми, що визначають купівельну поведінку клієнтів;

навчання персоналу виявлення покупців-машин.

Аналітики прогнозують, що до 2026 року клієнти-машини будуть здатні самостійно вибирати з конкуруючих продуктів потрібний власнику, а до 2036 року – робити самостійний вибір різних продуктів, аналізуючи потреби господаря.

Розширена підключена робоча сила. Генеративний штучний інтелект змінить те, як компанії організують робочі місця та розподіляють обов'язки. На думку аналітиків, це завдання вирішуватимуть генеративні мікрододатки – технологія, яка дозволить організаціям продемонструвати цінність генеративного ШІ, мінімізуючи при цьому ризики для бізнесу. Вони будуть працювати як проксі-сервер між користувачем та великою мовною моделлю, наприклад ChatGPT або Bard.

Gartner прогнозує, що до 2026 року 50% офісних працівників у компаніях зі списку Fortune 100 будуть у тій чи іншій формі працювати з ШІ або для підвищення продуктивності, або для покращення якості роботи. Наприклад, коли автор складатиме проект нового дослідження, мікрододаток у програмі обробки тексту активуватиме вбудовану бібліотеку підказок, щоб запросити у ШІ приклади підтверджуючих досліджень та даних, а також приклади суперечливих досліджень.

1.3 Огляд публікацій

Фізичний світ з розвитком кіберпростору перетворюється з пануючого в підлеглий. Всі фізичні процеси і явища мають власні цифрові образи, які поступово трансформуються в прообрази, а реальний світ стає все більш вразливим, залежним і керованим від віртуального кіберсвіту. Кіберфізічний світ з'єднує всіх мешканців планети один з одним без посередників, завдяки соціальним мережам, хмарним сервісам і Edge Computing.

Бібліотека IEEE Xplore сьогодні налічує кілька сотень джерел за напрямком Cyber Social Computing. Видавництво Springer має 6898 робіт. Можна прогнозувати, що поєднання двох ринково-орієнтованих наукових напрямків може дати істотний практичний результат для підвищення якості бізнесу, життя і збереження екології планети. Існує тільки одна книга видання Springer [3], як збірник статей за результатами однойменного наукового семінару, що побічно зачіпає питання управління кіберфізічним світом. Характерні публікації [4-6] орієнтовані на соціальні мережі і моніторинг переваг громадян без вироблення керуючих впливів в автоматичному режимі. Тому стаття [7], монографія [10], їх розвиток і вдосконалення, присвячені активному кіберфізичному комп'ютингу, пов'язаному з актюаторним управлінням соціальними, бізнес-процесами та явищами на основі їх точного моніторингу є дуже своєчасною і актуальною. Ринок сервісів в кіберпросторі поки по-старому використовує «печерні настінні» системи відображення інформації, призначені для очей людини, якій віддаються функції прийняття, як правило, помилкових актюаторних рішень, що призводять до соціальних колізій, економічним і фінансовим втратам. Позбавлення людини функції управління соціально-орієнтованим бізнесом і передача її кіберфізичному human-free бізнес-комп'ютингу є найголовнішою організаційною проблемою морального креативного світу. Людина не здатна точно управляти навіть сама собою, постійно забуваючи свій історичний досвід, вона регулярно наступає на «граблі» помилок минулого. Тому громадянин, соціальна група, компанія,

держава і людство потребують створення масштабованого аватара в форматі Gartner-computing: «virtual assistent – digital twin – smart robot», який позбавить людей від невірних рішень, що призводять до небажаних наслідків у бізнесі і на ринку соціальних технологій.

Основою класичного комп'ютингу традиційно є таблиця істинності [9-11], яка в явному вигляді задає функціональний опис поведінки технологічного об'єкта, робота, людини, соціальної групи. Інші форми, такі як: аналітичні рівняння, альтернативні графи, кубітно-векторні структури [10-12], орієнтовані на мінімізацію пам'яті для зберігання даних при описі складних функціональностей. Історія питання полягає у вичерпної формалізації обчислень, які пов'язані з процесами, що мають відношення до техніки і технологій. Щодо формального опису діяльності людини або соціальних груп, тут існує практично біла пляма, яка частково покривається машинним навчанням [13-16]. При синтезі таблиці істинності існує дві фази [13, 14]. Перша називається навчанням (Analysis), коли кожному вхідному (соціальному) впливу (Input Data Flow) ставиться в відповідність значення виходу (Truth Table) на підставі введеного критерію кластеризації (подібності-відмінності). Друга – тестування, коли виконується порівняння (Comparison) навченої або синтезованої таблиці істинності з фрагментом вже відомою. Результатом тестування є оцінка якості отриманої моделі (рис. 1.3), як кількість збігів G, относенное до загальної кількості тестових впливів (Test Table): Q = G / T.

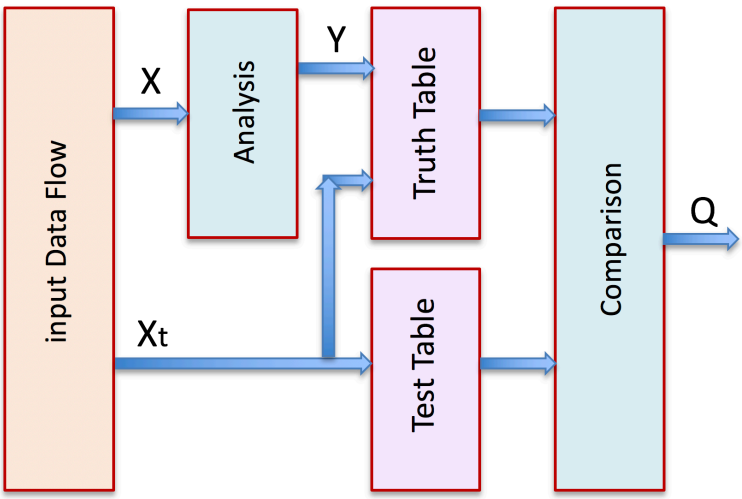


Рисунок 1.3 – Machine Learning for Truth Table Synthesis

Таким чином, для організації кіберсоціального комп'ютингу необхідно насамперед навчитися автоматично будувати таблиці істинності, які являють собою дискретний опис соціальних процесів і явищ, що призначені для моделювання і передбачення наслідків від прийняття рішень соціальними групами, керівниками та співробітниками компаній і університетів [17-21].

Одним з яскравих представників memory-driven social-комп'ютингу виступає компанія Wave Computing Inc, Silicon Valley, яка з'єднує AI-ML з dataflow-based MIPS RISC (конвеєр з обмеженою системою команд) architecture. Девіз компанії – "follow the data with AI" в цілях надання deep learning по будь-якому місцю знаходження великих даних – від дата центрів до терміналів хмари [22].

1.4 Висновки до розділу 1

Проналізовано тенденції сучасних електронних технологій на основі огляду літературних джерел та прогнозів американської дослідницької та консалтингової компанії Gartner.

Таким чином, на підставі огляду літературних джерел, сучасних технологічних тенденцій визначено актуальність дослідження.

Мета дослідження – суттєве зниження часу верифікації цифрових схем за рахунок векторного паралельного моделювання несправностей як адрес.

Для досягнення поставленої мети необхідно вирішити такі задачі:

– проаналізувати сучасні технологічні тенденції;

– виконати аналітичний огляд методів моделювання несправностей;

– вдосконалити автомат векторно-дедуктивного моделювання несправностей логіки.

Об'єкт дослідження – технології синтезу, аналізу, цифрових логічних структур для моделювання несправностей.

Предмет дослідження – моделі, методи і алгоритми синтезу та аналізу цифрових логічних структур.

2 ВИЗНАЧЕННЯ ТА МЕТРИКИ КОМП’ЮТИНГУ

Розглядаються визначення та метрики, орієнтовані на створення комп’ютингу, який поєднує підходи комп’ютерної інженерії та електронних технологій з соціальними процесами.

2.1 Визначення комп'ютингу та його похідних

Комп'ютинг – галузь знань, що займається розвитком теорії та практики надійного цифрового управління віртуальними, фізичними та соціальними процесами та явищами на основі вичерпного метричного моніторингу кіберфізичного простору шляхом використання cloud-edge сервісів та розумних сенсорів для збору та інтелектуальної обробки великих даних.

Комп'ютинг (системно) – цілеспрямований обчислювальний процес на основі цифрового моніторингу та управління механізмом виконання в метриці параметрів восьми параметрів: CEMORSAG (Control, Execution, Mode (Laws), Observation, Resources, State, Activation, Goal) (рис. 2.1).

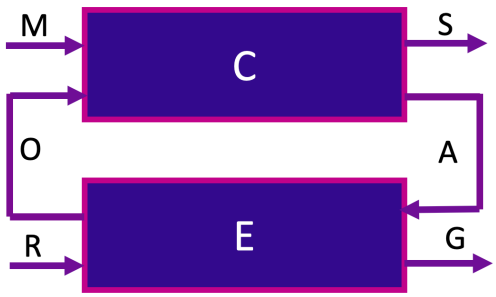


Рисунок 2.1 – Комп’ютинг у метриці CEMORSAG

Комп'ютинг (примітивно) – обчислювальний процес виконання транзакцій запису-зчитування даних на пам'яті, що адресується. Таке визначення на найнижчому рівні прибирає теоретичні бар'єри, створювані

теоремою Посту та повнотою функціонального базису [9], усуває логіку та шини передачі між процесором і пам'яттю в класичному комп'ютері. Крім того, транзакційний комп'ютинг усуває квантову логіку, що створює технологічні проблеми під час створення квантового комп'ютера. Для нового Q-комп'ютера достатньо реалізувати фотонні транзакції запису-зчитування даних на стійкій структурі атомарних електронів, що виконуються зі швидкістю світла.

Кіберфізичний простір – телекомунікаційна інфраструктура cloud-edge сервісів і розумних сенсорів, що об'єднує сукупність адресованих, метрично взаємодіючих, оцифрованих, віртуальних і реальних процесів і явищ з вираженими функціями моніторингу, обчислення, зберігання, транзакцій і управління.

Метрика – спосіб вимірювання відстаней чи відносин між компонентами процесів чи явищ, де виконується аксіома згортки у нуль циклічних відстаней (рис. 2.2): $\bigoplus_{i=\overline{1,n}} A_i = \ \cap A_i$.

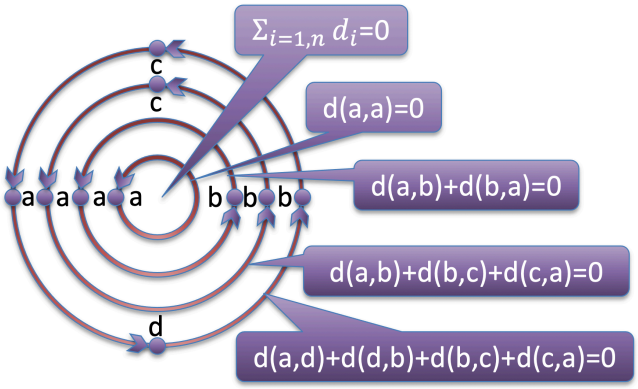


Рисунок 2.2 – Інфраструктура ML-cloud-edge сервісів

Відносини включення між компонентами укладу, що існує в суспільстві, визначається наступними ієрархічними зв’язками між моделлю, алгоритмом, методом, підходом, системою, укладом представлені на рис. 2.3. У цьому важливо знати визначення всіх елементарних понять, які входять в уклад.

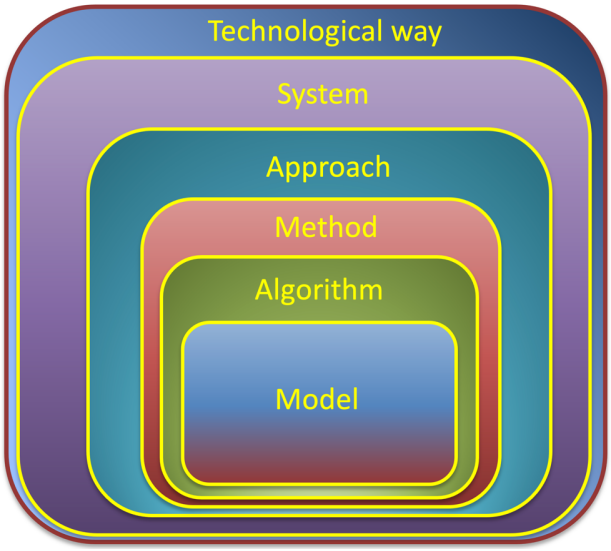


Рисунок 2.3 – Відносини включення між компонентами укладу

Модель (структура, алгебра) – сукупність відносин між компонентами, із заданою адекватністю описує властивості процесу чи явища.

Алгоритм – впорядкована у часі послідовність дій задля досягнення бажаного результату.

Метод – кінцева сукупність дій у просторі та часі для досягнення мети.

Підхід – система відносин, що інтегрує сукупність методів для дослідження та перетворення процесів та явищ.

Технологія – практичне застосування знання, методів та техніки у виробничій діяльності».

Система – сукупність відносин між компонентами та зовнішнім середовищем із вираженими функціями моніторингу та управління для досягнення поставлених цілей.

Уклад – якісний визначник розвитку науки і технологій та техніки, який задає на кілька десятиліть тренд розвиток усіх сфер людської діяльності, включаючи освіту, індустрію, транспорт, медицину, відпочинок та розваги. Тут можна говорити про наступні уклади: винахідництва (1900-1930), технічний (1930-1960), автоматизації (1960-1990), комп'ютера (1990-2020), цифрового інтелекту (2020-2050), рис. 2.4.

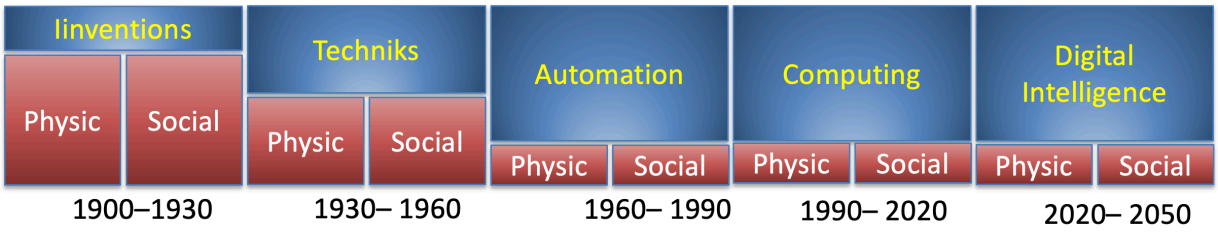


Рисунок 2.4 – Зміна технологічних укладів у новітній історії

З точки зору системного інженера, нову історію можна розділити на три епохи, якщо розглядати її як фізичний та соціальний світу, а також соціальне суспільство та його розвиток. Перший період характеризується розвитком кібернетики з незначними кроками до техніки та соціології. Другий період характеризується постійним прогресом цифровізації в усіх сферах людської діяльності, включаючи технології та соціологію (рис. 2.5). Третю еру слід вважати чисто аналоговою або квантовою, в якій людство спілкується образно і телепатично за допомогою мозку, і вирішуються всі питання, що стосуються біологічного існування, суспільства і морального лідерства кожної людини.

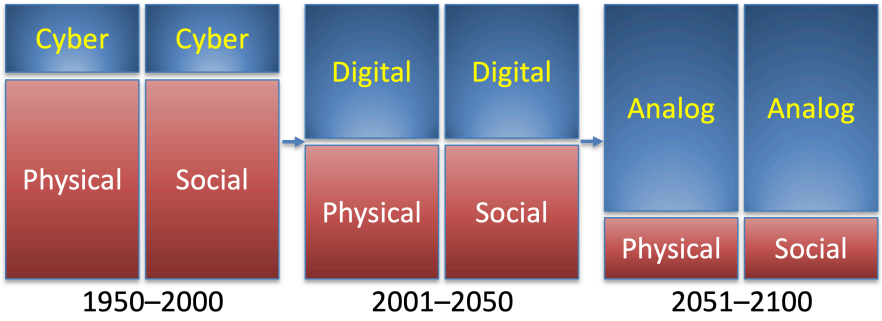


Рисунок 2.5 – Зміна технологічних укладів у новій історії

Відношення – причина розвитку природи та суспільства. Всесвіт чи природа (суспільство) є відношенням і нічого, крім цього. Неймовірне є відношення між очевидними процесами та явищами в оригінальній структурі. Будь-яке відношення є результатом (помилка) гри між ідеалом і реальністю: F⊕T=L (рис. 2.6). У цю схему укладається процес навчання ML-моделі T і

процес удосконалення специфікації F=L⊕T, що призводить до принципово нового рішення генеративного машин ленінг комп'ютинга.

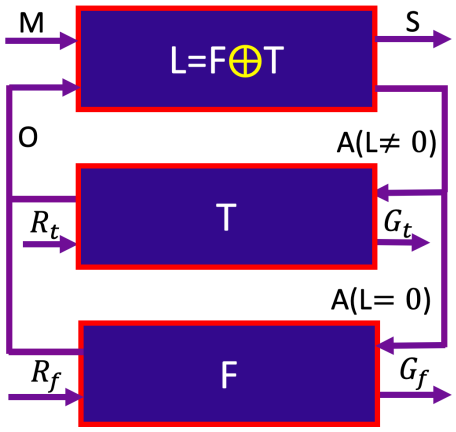


Рисунок 2.6 – Excellence Relations Computing

Дискретна математика – відносини між структурами даних та алгоритмами. Нині вона у розквіті. Аналогова математика тимчасово здає свої позиції, але її майбутнє квантового компьютинга. Штучний інтелект – відношення між точною специфікацією та наближеним імовірнісним рішенням. Комп'ютинг – відношення між механізмами управління та виконання для досягнення мети. Комп'ютерна інженерія – відносини між апаратними та програмними рішеннями для досягнення мети. Квантовий комп'ютинг – відносини між механізмами цифрового управління та аналогового виконання. Де немає природного інтелекту, там з'являється штучний. Природний інтелект створює точне рішення за мінімальний час, а штучний – ймовірнісний за максимальний час. Цифровий комп'ютинг з'явився 40 тисяч років тому, з першим малюнком мамонта на стіні в печері. Квантовий космологічний комп'ютинг існує з моменту народження Всесвіту або завжди.

Failure-driven computing у метриці T⊕F⊕L=0 здатний найбільш формально уявити сутність будь-якого комп'ютингу на основі xor-відносин між трьома компонентами, два з яких відомі (рис. 2.7).

Failure-driven T⊕ F⊕L=0 computing		
Kind of computing	Equation	Description
Testing	L= T⊕F	L – faults, T – test, F – function
Design and test	T= F⊕L	T – SoC, F – specification, L – faults
Computing	T= F⊕L	T – computer, F – specification, L – failure
Computer engineering	T= F⊕L	T – computers, F – theory, L – contradiction
Quantum computing	T= F⊕L	T –realization,F –specification, L –inaccuracies
Artificial intelligence	T= F⊕L	T – realization, F – specification, L – error
Social computing	L= F⊕T	L – criminal, F – legislations, T – traditions
Cyber social computing	F= T⊕L	F – legislations, T – traditions, L – violations
Federated ML Computing	T= F⊕L	T – ML-model, F – specification, L – error
Generative MLComputing	F= T⊕L	F – novelty L – failure, T – existent
Cosmological computing	F= T⊕L	F – novelty L – Failure, T – Universum

Рисунок 2.7 – Failure-driven computing у метриці T⊕F⊕L=0

Природно, що ця метрика може бути продовжена на будь-який керований обчислювальний процес на основі вичерпного моніторингу: кіберсоціальний, охорона здоров'я, транспорт, наука, освіта, виробництво, банкінг, торгівля, відпочинок, туризм, розвага, культура, мистецтво, економіка, політика, побут, харчування, спорт, державне управління громадянами. Скрізь є помилки, отже – шлях до досконалості.

2.2 Векторна метрика опису процесів та явищ

У технічній діагностиці використовуються три основних формати опису процесів і явищ: табличний, аналітичний і графічний [9, 10, 23-28]. При цьому

матриці (таблиці) і вектори є двома формами опису моделей, які поєднуються між собою. Вектор (двійковий, багатозначний) – це компактна форма таблиці істинності у вигляді впорядкованої послідовності вихідних станів, коли адреси вхідних компонент розташовані в порядку зростання [9, 26, 31, 33]. При необхідності матриці розкладаються на одновимірні вектори для спрощення паралельної обробки даних в регістрової пам'яті. Звичайно, з векторних форм опису процесів і явищ дуже легко реконструювати таблиці та матриці. Далі вектори з'являються як формат опису об'єкта. Метрики для вимірювання процесів і явищ у дискретному (бінарному) просторі використовує три аксіоми періодичних або замкнутих взаємодій між одним, двома та трьома компонентами: рефлексивність, симетрія та транзитивність.

Для загального випадку виміру застосовується універсальний закон конволюції відстаней між n об'єктами, що становлять цикл: $\bigoplus_{i=1}^{n} d_i=0$ [10]. При цьому xor-сума відстаней між замкнутими функціональними об'єктами, поданими як стани вихідної змінної, також завжди дорівнює нулю. XOR-сума кубітних векторів функцій, як об'єктів (and = 0001, or = 01111, xor = 0110), дорівнює нулю (рис. 2.8).

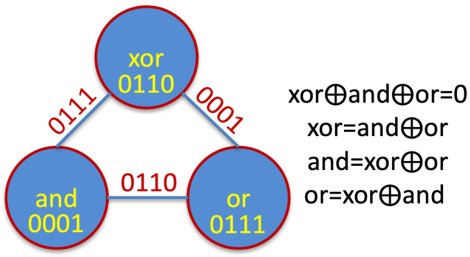


Рисунок 2.8 – Конволюція трьох логічних функцій

Таким чином виходить, що xor-відношення двох згаданих функцій між собою дорівнюють третій. Спільно три функції (Xor, And, Or) у вигляді xor–відносини згортаються в нуль: X⊕A⊕O=0. Вірно також, що дві з трьох функцій генерує третю, як xor-відношення між ними.

Лемма конволюції трьох функцій від двох змінних: існують лише 4 пари логічних функцій, пов'язаних xor-відношенням, які в результаті дають xor-функцію:

1	2	3	4
$\oplus$=0110	$\oplus$=0110	$\oplus$=0110	$\oplus$=0110
$\wedge$=0001	$\overline{\vee}$=1000	X_1=0011	$\overline{X}_1$=1100
$\vee$=0111	$\overline{\wedge}$=1110	X_2=0101	$\overline{X}_2$=1010

Тут перші два варіанти фактично можна розглядати як пару логічних функцій, взаємодія яких дає функцію xor. Дві інші є виродженими функціями змінних, які створюють операцію xor. Враховуючи, що другий варіант є зворотним до першого варіанту, можна дистати висновку, що лише дві логічні функції (і, або) можуть створити вектор xor із взаємодіями xor.

Практичний зміст леми про функцію згортки полягає в наступному:

1) оператор xor є унікальною універсальною мірою подібності та відмінності між усіма кіберфізичними процесами та явищами;

2) лише дві логічні функції (і, або) виробляють операцію XOR через свою різницю;

3) дві функції з однієї змінної (інверсія та повторення) також створюють взаємодію xor;

4) тріада логічних функцій (and, or, xor) формує векторну метрику для обчислення структурних: $S(a,b) = a_i \underset{i=1,n}{\wedge} b_i$, $D(a,b) = a_i \underset{i=1,n}{\oplus} b_i$, і нормованих оцінок подібності-відмінності між процесами та явищами:

$$S_n(a,b) = \frac{\Sigma_{i=1}^{n}\left(a_i \underset{i=1,n}{\wedge} b_i\right)}{\Sigma_{i=1}^{n}\left(a_i \underset{i=1,n}{\vee} b_i\right)}, D_n(a,b) = \frac{\Sigma_{i=1}^{n}\left(a_i \underset{i=1,n}{\oplus} b_i\right)}{\Sigma_{i=1}^{n}\left(a_i \underset{i=1,n}{\vee} b_i\right)};$$

5) операція диз'юнкції (or) створює загальну метрику одиничних значень координат для виміру норми подібності-відмінності двох процесів чи явищ.

2.3 Висновки до розділу 2

Наведено система понять, визначень та метрики, що орієнтовані на створення комп’ютингу, який поєднує підходи комп’ютерної інженерії та електронних технологій з соціальними процесами.

3 МЕТОДИ МОДЕЛЮВАННЯ НЕСПРАВНОСТЕЙ

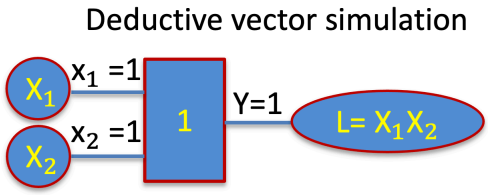
Аналізується математичний апарат – методи дедуктивного моделювання несправностей: дедуктивно-векторний; метод векторного дедуктивного аналізу логічних структур; векторний (кубітний) метод синтезу дедуктивних функцій; синтез дедуктивних моделей основних логічних елементів.

3.1 Дедуктивно-векторний метод моделювання несправностей

Дедуктивне моделювання, запропоноване рівно 50 років тому Армстронгом, досі є найвитонченішим і найефективнішим засобом аналізу якості тестів та синтезу таблиць для пошуку дефектів. Крім того, даний метод може бути ефективно використаний для аналізу виборчої активності шляхів прийняття рішень будь-якої кібер-соціальної системи, яка представлена елементами логіки.

Реалізація на основі векторної форми опису логіки запропонована у роботах [25, 27, 30, 31, 32] та надає можливість суттєво спростити алгоритми дедуктивного моделювання з метою обробки цифрових схем великої розмірності.

Трикутник xor–відносин є основою та використовується для векторної модифікації дедуктивного методу моделювання несправностей. Новий метод синтезу дедуктивних формул з урахуванням векторного завдання, у разі or–елемента наведено на рис. 3.1.



Q=0111 or				
x_1x_2	00	01	10	11
$\hat{x}_1$	0011	0011	0011	0011
$\hat{x}_2$	0101	0101	0101	0101
Q	0111	0111	0111	0111
$\vec{x}_1$=$\hat{x}_1 \oplus x_1$	0011	0011	1100	1100
$\vec{x}_2$=$\hat{x}_2 \oplus x_2$	0101	1010	0101	1010
$\vec{L}$=$Q \oplus$Y	0111	1000	1000	1000
L	$X_1 \vee X_2$	$\bar{X}_1X_2$	$X_1\bar{X}_2$	X_1X_2

Рисунок 3.1 – Дедуктивно-векторне or-logic моделювання

Двійкові вектори обробляються за такими правилами обробки їх координат:

$$\overrightarrow{x_i} = \hat{x}_2 \oplus x_i, x_i = \{0,1\};$$
$$\overrightarrow{x_1} = \hat{x}_1 \oplus x_1 = 0, \overrightarrow{x_1} = \hat{x}_1 \oplus 0 = 0011 \oplus 0 = 0011;$$
$$\overrightarrow{x_1} = \hat{x}_1 \oplus x_1 = 1, \overrightarrow{x_1} = \hat{x}_1 \oplus 1 = 0011 \oplus 1 = 1100;$$
$$\overrightarrow{x_2} = \hat{x}_2 \oplus x_2 = 0, \overrightarrow{x_2} = \hat{x}_2 \oplus 0 = 0011 \oplus 0 = 0011;$$
$$\overrightarrow{x_2} = \hat{x}_2 \oplus x_2 = 1, \overrightarrow{x_2} = \hat{x}_2 \oplus 1 = 0011 \oplus 1 = 0011.$$

Тут вводиться нова оригінальна модель одиночних несправностей, прив'язана до координат векторів-стовпців, що представляють вектор-компоненти моделі логічного елемента:

$$\hat{x}_1 = 0011 \hat{x}_2 = 0101, Q = 0111,$$

отримані зі стовпців наступної таблиці істинності (табл. 3.1):

Таблиця 3.1 – Таблиця істинності логічного елемента

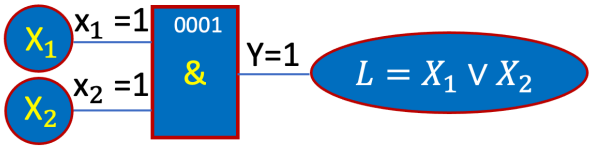
$\hat{x}_1$	$\hat{x}_2$	Q
0	0	0
0	1	1
1	0	1
1	1	1

При цьому список-вектор дефектів, що перевіряються, формується шляхом виконання xor-операції між функціональністю і тестовим набором, які також задаються у векторному вигляді, наприклад:

$$\text{L=T}\oplus F = 01010101 \oplus 10001111 = 10101010.$$

У координатах дедуктивної таблиці представлені процедури виконання класичного дедуктивного алгоритму для транспортування дефектів на вихід логічного елемента, але при описі функціональностей та дефектів у векторному вигляді. Задаються функції змінних x_1, x_2 у векторному вигляді $\hat{x}_1 = 0011$, $\hat{x}_2 = 0101$. Потім виконується інверсія координат векторів, якщо стан вхідної змінної дорівнює одиниці. Те саме стосується вектора вихідний змінний Q: $\vec{L}$=$Q\oplus$Y, за умови Y=f(x_1, x_2). Потім за константами одиниці даного вектора $\vec{L}$ виконується запис термів диз'юнктивної нормальної форми L щодо змінних векторів $\hat{x}_1$ и $\hat{x}_1$. Якщо L=$X_1 \vee X_2$, це означає, що вихідний список несправностей логічного елемента міститиме об'єднання вхідних списків несправностей.

Список-вектор L дефектів, що транспортуються на вихід, записується ДНФ вхідних списків несправностей, представлених в останньому рядку таблиці. Синтез дедуктивних формул на основі векторних операцій для отримання ДНФ для логічного елемента and з метою транспортування векторів або списків вхідних несправностей представлено на рис. 3.2.



Q=0001 and				
x_1x_2	00	01	10	11
$\hat{x}_1$	0011	0011	0011	0011
$\hat{x}_2$	0101	0101	0101	0101
Q	0001	0001	0001	0001
$\vec{x}_1=\hat{x}_1\oplus x_1$	0011	0011	1100	1100
$\vec{x}_2=\hat{x}_2\oplus x_2$	0101	1010	0101	1010
$\vec{L}=Q\oplus Y$	0111	1000	1000	1000
L	X_1X_2	$\bar{X}_1X_2$	$X_1\bar{X}_2$	$X_1 \vee X_2$

Рисунок 3.2 – Дедуктивно-векторне and-logic моделювання

Для порівняння обчислювальної складності отримання дедуктивних формул нижче наводиться аналітична класична процедура аналізу або елемента, що оперує булевими рівняннями, для яких необхідно мати складні вирішувачі [26, 31, 33]:

$$
\begin{aligned}
&L[T=(00,01,10,11),F=(X_1\vee X_2)]=\\
&=L\{(\bar{x}_1\bar{x}_2\vee\bar{x}_1x_2\vee x_1\bar{x}_2\vee x_1x_2)\wedge[(X_1\oplus T_{t1}\vee X_2\oplus T_{t2})\oplus T_{t3})]\}=\\
&=(\bar{x}_1\bar{x}_2)\{[(X_1\oplus 0)\vee(X_2\oplus 0)]\oplus 0\}\vee(\bar{x}_1x_2)\{[(X_1\oplus 0)\vee(X_2\oplus 1)]\oplus 1\}\vee\\
&\vee(x_1\bar{x}_2)\{[(X_1\oplus 1)\vee(X_2\oplus 0)]\oplus 1\}\vee(x_1x_2)\{[(X_1\oplus 1)\vee(X_2\oplus 1)]\oplus 1\}=\\
&=(\bar{x}_1\bar{x}_2)(X_1\vee X_2)\vee(\bar{x}_1x_2)(\bar{X}_1\wedge X_2)\vee(x_1\bar{x}_2)(X_1\wedge\bar{X}_2)\vee(x_1x_2)(X_1\wedge X_2).
\end{aligned}
$$

Природно, що обробка векторних моделей є більш технологічною процедурою, де обчислювальна складність отримання дедуктивних формул транспортування дефектів функції від n змінних дорівнює Q = n(N+F). Якщо врахувати, що N+F, як процедури інверсії векторів N та їх подальшого складання F у вектор-функцію (об'єднання або перетин) дорівнюють числу змінних N+F=n, то оцінка обчислювальної складності у векторних операціях дорівнює $Q = n(N+ F) = n^2$.

3.2 Дедуктивний аналіз логічної схеми

Дано цифрову структуру з чотирьох елементів, кожен з яких представлений Q-вектором опису станів вихідної змінної Y у відповідній таблиці істинності (рис. 3.3).

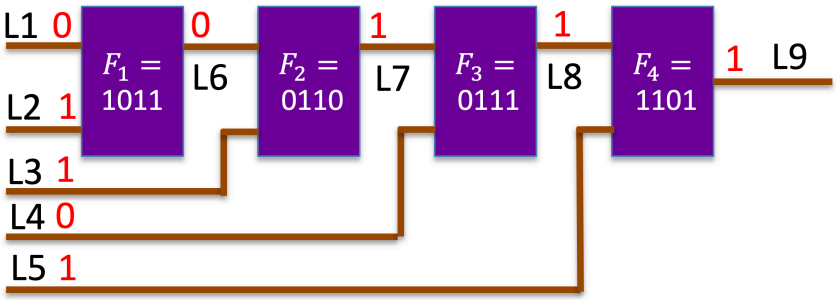


Рисунок 3.3 – Цифрова структура із чотирьох елементів

Завдання полягає в тому, щоб на вхідному двійковому наборі X=01101 визначити список вхідних несправностей, які будуть транспортовані від кожного входу схеми L={L1,L2,L3,L4,L5,} до виходу L9 = F[X, L(x_i)].

Для вирішення цієї задачі необхідно побудувати диз'юнктивні нормальні форми транспортування вхідних списків несправностей через кожен елемент схеми.

Таблиця 3.1 містить векторні процедури обробки Q векторів елементів для отримання дедуктивних формул на кожну двійкову дію.

Враховуючи регулярність даних у таблицях та повторення векторних процедур, можна інтегрувати всі чотири таблиці в одну (табл. 3.2), де перші три рядки однакові для всіх елементів, а потім йдуть 5 рядків різноманітності, пов'язані з особливостями поведінки елементів на вхідних наборах.

Таблиця 3.1 – Векторні процедури обробки Q векторів елементів

Q=1011 F1				
x_1x_2	00	01	10	11
$\hat{x}_1$	0011	0011	0011	0011
$\hat{x}_2$	0101	0101	0101	0101
Q	1011	1011	1011	1011
$\vec{x}_1=\hat{x}_1 \oplus x_1$	0011	0011	1100	1100
$\vec{x}_2=\hat{x}_2 \oplus x_2$	0101	1010	0101	1010
$\vec{L}=Q \oplus Y$	0100	1011	0100	0100
L	$\bar{X}_1X_2$	$X_1 \vee X_2$	$\bar{X}_1X_2$	$\bar{X}_1X_2$

Q=0110 F2				
x_1x_2	00	01	10	11
$\hat{x}_1$	0011	0011	0011	0011
$\hat{x}_2$	0101	0101	0101	0101
Q	0110	0110	0110	0110
$\vec{x}_1=\hat{x}_1 \oplus x_1$	0011	0011	1100	1100
$\vec{x}_2=\hat{x}_2 \oplus x_2$	0101	1010	0101	1010
$\vec{L}=Q \oplus Y$	0110	1001	1001	0110
L	$X_1\bar{X}_2 \vee \bar{X}_1X_2$			

Q=0111 F3				
x_1x_2	00	01	10	11
$\hat{x}_1$	0011	0011	0011	0011
$\hat{x}_2$	0101	0101	0101	0101
Q	0111	0111	0111	0111
$\vec{x}_1=\hat{x}_1 \oplus x_1$	0011	0011	1100	1100
$\vec{x}_2=\hat{x}_2 \oplus x_2$	0101	1010	0101	1010
$\vec{L}=Q \oplus Y$	0111	1000	1000	1000
L	$X_1 \vee X_2$	$\bar{X}_1X_2$	$X_1\bar{X}_2$	X_1X_2

Q=1101 F4				
x_1x_2	00	01	10	11
$\hat{x}_1$	0011	0011	0011	0011
$\hat{x}_2$	0101	0101	0101	0101
Q	1101	1101	1101	1101
$\vec{x}_1=\hat{x}_1 \oplus x_1$	0011	0011	1100	1100
$\vec{x}_2=\hat{x}_2 \oplus x_2$	0101	1010	0101	1010
$\vec{L}=Q \oplus Y$	0010	0010	1101	0010
L	$X_1\bar{X}_2$	X_1X_2	$X_1 \vee X_2$	$\bar{X}_1X_2$

Таблиця 3.2 – Інтегрована таблиця 3.1

	x_1x_2	00	01	10	11
	$\hat{x}_1$	0011	0011	0011	0011
	$\hat{x}_2$	0101	0101	0101	0101
F1	Q	1011	1011	1011	1011
	$\vec{x}_1=\hat{x}_1 \oplus x_1$	0011	0011	1100	1100
	$\vec{x}_2=\hat{x}_2 \oplus x_2$	0101	1010	0101	1010
	$\vec{L}=Q \oplus Y$	0100	1011	0100	0100
	L	$\bar{X}_1X_2$	$X_1 \vee X_2$	$\bar{X}_1X_2$	$\bar{X}_1X_2$
F2	Q	0110	0110	0110	0110
	$\vec{x}_1=\hat{x}_1 \oplus x_1$	0011	0011	1100	1100
	$\vec{x}_2=\hat{x}_2 \oplus x_2$	0101	1010	0101	1010
	$\vec{L}=Q \oplus Y$	0110	1001	1001	0110
	L	$X_1\bar{X}_2 \vee \bar{X}_1X_2$			
F3	Q	0111	0111	0111	0111
	$\vec{x}_1=\hat{x}_1 \oplus x_1$	0011	0011	1100	1100
	$\vec{x}_2=\hat{x}_2 \oplus x_2$	0101	1010	0101	1010
	$\vec{L}=Q \oplus Y$	0111	1000	1000	1000
	L	$X_1 \vee X_2$	$\bar{X}_1X_2$	$X_1\bar{X}_2$	X_1X_2
F4	Q	1101	1101	1101	1101
	$\vec{x}_1=\hat{x}_1 \oplus x_1$	0011	0011	1100	1100
	$\vec{x}_2=\hat{x}_2 \oplus x_2$	0101	1010	0101	1010
	$\vec{L}=Q \oplus Y$	0010	0010	1101	0010
	L	$X_1\bar{X}_2$	X_1X_2	$X_1 \vee X_2$	$\bar{X}_1X_2$

Далі можна стиснути інформацію в цій таблиці до п'яти рядків (табл. 3.3), залишивши лише дедуктивні формули обробки вхідних списків, залежно від вхідного значення на кожному елементі.

Таблиця 3.3 – Стиснута таблиця 3.2

	x_1x_2	00	01	10	11
F1	L	$\bar{X}_1X_2$	$X_1 \vee X_2$	$\bar{X}_1X_2$	$\bar{X}_1X_2$
F2	L	$X_1\bar{X}_2 \vee \bar{X}_1X_2$			
F3	L	$X_1 \vee X_2$	$\bar{X}_1X_2$	$X_1\bar{X}_2$	X_1X_2
F4	L	$X_1\bar{X}_2$	X_1X_2	$X_1 \vee X_2$	$\bar{X}_1X_2$

Таблиця 3.3 використовується для моделювання будь-яких векторів списків несправностей L={L1,L2,L3,L4,L5,} залежно від вхідного двійкового слова з метою отримання вихідного списку несправностей L9. Результат дедуктивного моделювання векторів L на цифровій структурі логічних елементів наведено в таблиці 3.4.

Таблиця 3.4 – Результат дедуктивного моделювання векторів

	x_1x_2	00	01	10	11
F1	L1=01101110 L2 =11111001 L6 =01101110	$\bar{X}_1X_2$	$X_1 \vee X_2$	$\bar{X}_1X_2$	$\bar{X}_1X_2$
F2	L6=11111001 L3=01101110 L7=10010111	$X_1\bar{X}_2 \vee \bar{X}_1X_2$ D S D			
F3	L7=00000110 L4=11011011 L8=00000100	$X_1 \vee X_2$	$\bar{X}_1X_2$	$X_1\bar{X}_2$	X_1X_2
F4	L8=00000100 L5=10011110 L9=10011010	$X_1\bar{X}_2$	X_1X_2	$X_1 \vee X_2$	$\bar{X}_1X_2$

Кінцевий результат отримання дедуктивних формул та їх застосування транспортування векторів вхідних списків несправностей до виходу представлений на рис. 3.4. Елемент F2, дедуктивна формула якого представлена другим рядком таблиці має унікальну характеристику, яка

дозволяє йому пропускати через себе тільки відмінності (D), які є в його списках L= $X_1\bar{X}_2 \vee \bar{X}_1X_2$. При цьому елемент поводиться однаково інваріантно до будь-якого двійкового вхідного слова. Чим не ідеальна модель чиновника-експерта, який дає дорогу всьому новому від будь-якого громадянина та фільтрує старі неефективні рішення (S) незалежно від рангу подавця.

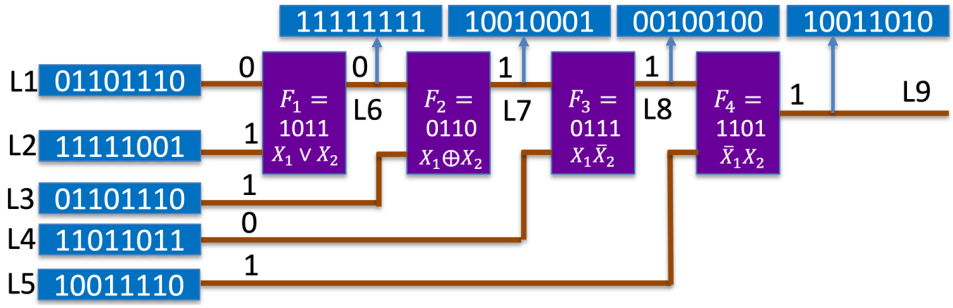


Рисунок 3.4 – Моделювання чотирьох елементної цифрової структури

Впровадження векторно-дедуктивного методу до системи автоматизації проектування цифрових схем дозволить суттєво спростити методи оцінки якості тестів, а також зменшити час проходження проекту при оцінці його якості за рахунок векторних паралельних процедур транспортування списків несправностей до виходів схеми.

Кіберсоціальна інтерпретація цього результату полягає в тому, що кожній змінній схеми можна поставити у відповідність деякого чиновника, який дозволяє або гальмує проходження активності на вихід соціально-логічної структури управління. При цьому роль кожного елемента такої ієрархії полягає у фільтрації-посиленні активності, що йде від громадян, яка може відігравати як позитивну, так негативну роль. В результаті, зокрема, на виході схеми може бути нуль активностей при потужних вхідних списках, а може бути ситуація посилення активностей, що знаходяться на вході схеми (збільшення їх потужності). Конструктивний менеджер перетворює відсутність активностей при майже нульових вхідних списках на деякі досягнення – збільшення активностей на виході схеми кіберсоціального комп'ютингу. Дедуктивна логіка це дозволяє.

Таким чином, метод векторного дедуктивного аналізу логічних структур відрізняється від відомих технологій моделювання [26, 29, 30, 31, 32, 36] високою структуризацією векторних даних, найпростішими логічними паралельними процедурами та високою швидкодією отримання дедуктивних формул та моделювання логічних елементів, що дає можливість його використання для обробки схем великої розмірності. Метод векторного дедуктивного аналізу логічних структур не має аналогів у світі про простоту, продуктивність, а його універсальність допускає його використання в різних сферах діяльності людини,

3.3 Векторний (кубітний) метод синтезу дедуктивних функцій

Метод орієнтований на логіку довільної розмірності, яка задається кубітним вектором функціональності, що робить його практично орієнтованим з огляду на технологічність і простоту реалізації. Як правило, методи синтезу дедуктивних формул для цифрових схем залишаються поза публікаціями, які описують використання вже готових логічних виразів. Далі розглядається підхід отримання таких формул, а точніше матриць для дедуктивного аналізу цифрових схем. Найпростішою та найефективнішою технологією синтезу дедуктивних функцій для моделювання несправностей є метод, що використовує кубітні або векторні покриття булевих функцій. Тут під кубітним Q-покриттям розуміється вектор станів вихідний змінної таблиці істинності [10, 26, 34, 35], де кожній координаті вектора ставиться у відповідність двійкову адресу, ототожнюється із вхідним набором функціонального елемента. Якщо зважити на той факт, що для кожного вхідного набору існує власна дедуктивна функція, то для всіх вхідних послідовностей необхідно побудувати матрицю розмірністю $\mathbf{2^n \times 2^n}$. Процесор синтезу дедуктивної матриці для паралельного аналізу одиночних константних несправностей на основі Q-покриття функціональності містить тільки дві операції (рис. 3.5).

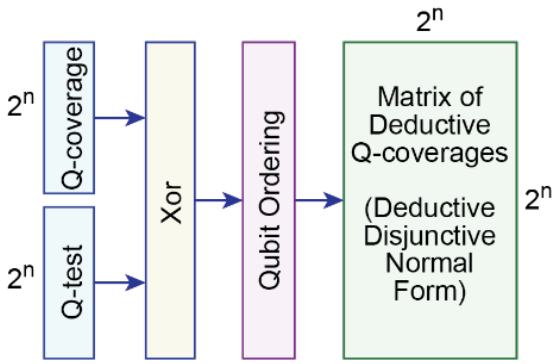


Рисунок 3.5 – Секвенсор синтезу векторів дедуктивної матриці

1) Синтез матриці Q-векторів дедуктивних функцій залежить від двійково-десяткового номера вхідного набору та координат Q-вектора для функції від змінних n: $L^{*}_{ij} = (Q_i \underset{i=\overline{1,m}}{\oplus} Q_j)_{j=1}^{m}$, $m=2^n$.

Фактично береться перша координата кубитного покриття (вектора), яка xor-взаємодіє з усіма координатами Q-вектора функціональності для формування вектора дедуктивного відносно першої вхідної послідовності. Потім береться друга координата Q-вектора, яка xor-взаємодіє з усіма координатами Q-покриття. Процедура закінчується після того, як усі координати Q-вектора були xor-складені з кубітним вектором. Обчислювальна складність даної процедури дорівнює $\boldsymbol{2^{n+1}}$, яка може бути зменшена до n шляхом апаратної реалізації паралельного виконання xor-операції.

2) Після отримання дедуктивної матриці для всіх вхідних наборів по кубитному покриттю функціональності необхідно виконати процедуру перестановки бітів у стовпцях отриманої матриці за правилом: $L_{ij} = [L^{*}(H_{ij})]_{\{i,j\}=1}^{m}$ відповідно до номерів, представлених у матриці перестановки H, яка за розмірністю дорівнює матриці кубитних покриттів дедуктивних функцій. Далі наведено приклади отримання матриці дедуктивних векторів та відповідних їм функцій для логіки (and, or, xor, not-xor, not, repeater) на основі використання тільки кубитних покриттів функціональностей [26, 31]:

Q_{and}	L_1	L_2	L_3	L_4
0	0	0	0	1
0	0	0	0	1
0	0	0	0	1
1	1	1	1	0

→

1	2	3	4
2	1	4	3
3	4	1	2
4	3	2	1

→

X_1X_2	L_1	L_2	L_3	L_4
00	0	0	0	0
01	0	0	1	1
10	0	1	0	1
11	1	0	0	1

→

$$\rightarrow L = (00)(X_1X_2) \vee (01)(X_1\bar{X}_2) \vee (10)(\bar{X}_1X_2) \vee (11)(X_1 \vee X_2).$$

Q_{or}	L_1	L_2	L_3	L_4
0	0	1	1	1
1	1	0	0	0
1	1	0	0	0
1	1	0	0	0

→

1	2	3	4
2	1	4	3
3	4	1	2
4	3	2	1

→

X_1X_2	L_1	L_2	L_3	L_4
00	0	0	0	0
01	1	1	0	0
10	1	0	1	0
11	1	0	0	1

→

$$\rightarrow L = (00)(X_1 \vee X_2) \vee (01)(\bar{X}_1X_2) \vee (10)(X_1\bar{X}_2) \vee (11)(X_1X_2).$$

Q_{xor}	L_1	L_2	L_3	L_4
0	0	1	1	0
1	1	0	0	1
1	1	0	0	1
0	0	1	1	0

→

1	2	3	4
2	1	4	3
3	4	1	2
4	3	2	1

→

X_1X_2	L_1	L_2	L_3	L_4
00	0	0	0	0
01	1	1	1	1
10	1	1	1	1
11	0	0	0	0

→

$$\rightarrow L = (00 \vee 01 \vee 10 \vee 11)(\bar{X}_1X_2 \vee X_1\bar{X}_2) = (xx)(\bar{X}_1X_2 \vee X_1\bar{X}_2).$$

Q_{nxr}	L_1	L_2	L_3	L_4
1	0	1	1	0
0	1	0	0	1
0	1	0	0	1
1	0	1	1	0

→

1	2	3	4
2	1	4	3
3	4	1	2
4	3	2	1

→

X_1X_2	L_1	L_2	L_3	L_4
00	0	0	0	0
01	1	1	1	1
10	1	1	1	1
11	0	0	0	0

→

$$\rightarrow L = (00 \vee 01 \vee 10 \vee 11)(\bar{X}_1X_2 \vee X_1\bar{X}_2) = (xx)(\bar{X}_1X_2 \vee X_1\bar{X}_2).$$

Q_{not}	L_1	L_2
1	0	1
0	1	0

→

1	2
2	1

→

X	L_1	L_2
0	0	0
1	1	1

→

$$\rightarrow L = (0 \vee 1)(X \vee X) = (xx)(X).$$

Q_{rep}	L_1	L_2
0	0	1
1	1	0

→

1	2
2	1

→

X	L_1	L_2
0	0	0
1	1	1

→

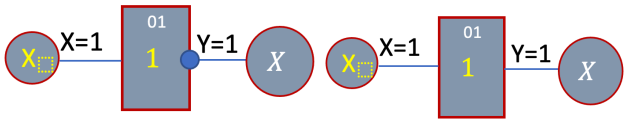
$$\rightarrow L = (0 \vee 1)(X \vee X) = (xx)(X).$$

Тут процедура перестановки бітів є рекурсивною та регулярною, яка не є обчислювально складною [26]. Запропонований метод синтезу дедуктивних матриць для моделювання несправностей відрізняється від відомих у світі аналогів оригінальністю математичних рішень, високим рівнем паралелізму та компактністю структур даних, що дозволяє використовувати його програмну або апаратну реалізацію для синтезу, аналізу, тестування, верифікації та діагностування цифрових систем на кристалах. Для розвитку memory-driven logic-free комп'ютера запропонований кубитно-векторний дедуктивний аналіз є істотним внеском у проектування ремонтопридатних цифрових систем регістрового та системного рівня опису [26, 31]. Висока продуктивність синтезу кубічних дедуктивних покриттів $Q=2\times 2^n = 2^{n+1}$, є підставою для

його використання в режимі online, оскільки реєстрова реалізація фактично двох операцій: xor-порівняння та перестановки бітів дозволить звести згадану оцінку до 2–5 автоматних тактів. Крім того, дві згадані операції можна поєднати в одну процедуру, що виконується в одному автоматному такті.

3.4 Синтез дедуктивних формул основних логічних елементів

Інтерес представляє також процес синтезу дедуктивних моделей основних логічних елементів, які можна використовувати як бібліотеку для створення та аналізу кіберсоціального управління схем соціальними процесами в університетах у компаніях та державі. Далі подано таблиці векторного синтезу дедуктивних формул для перевірки логічних схем управління соціумом. Найбільш примітивними елементами є інвертор та повторювач. Незважаючи на те, що це різні елементи, вони мають однакову дедуктивну формулу, яка дозволяє проводити цифрову чи соціальну активність на вході до виходу, не спотворюючи її (рис. 3.6).



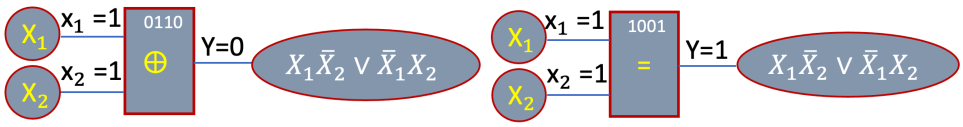
Q=01		
x	0	1
$\hat{x}$	01	01
$\vec{x}$	01	10
Q	01	01
$\vec{L}=Q\oplus Y$	01	10
L	X	

Q=10		
x	0	1
$\hat{x}$	01	01
$\vec{x}$	01	10
Q	10	10
$\vec{L}=Q\oplus Y$	01	10
L	X	

Рисунок 3.6 – Синтез дедуктивних формул для примітивних взаємно інверсних елементів: 10 та 01

У наступному кадрі показані процедури синтезу дедуктивних формул для пари логічних функцій 0110 та 1001, які є взаємно інверсними один одному (рис. 3.7). Пара чудових функцій має властивість, що на будь-якому одному

впливі вони фільтрують всі пасивності і пропускають через себе будь-яку активність, яка була присутня на їх входах. Їх можна використовувати як plagiarism-фільтр всього старого. Ці елементи здатні пропускати на вихід тільки оригінальні нові результати від будь-якого входу (громадянина) на будь-якому вхідному впливі чи формі правління. Ці елементи використовуються як універсальний цифровий вимірник всіх процесів і явищ у кіберпросторі, включаючи моральний моніторинг та управління соціальними групами.



Q=0110				
x_1x_2	00	01	10	11
$\hat{x}_1$	0011	0011	0011	0011
$\hat{x}_2$	0101	0101	0101	0101
$\vec{x}_1=\hat{x}_1\oplus x_1$	0011	0011	1100	1100
$\vec{x}_2=\hat{x}_2\oplus x_2$	0101	1010	0101	1010
$L_v=\vec{x}_1\oplus\vec{x}_2$	0110	1001	1001	0110
$\vec{L}=L_v\oplus Y$	0110	0110	0110	0110
L	$X_1\bar{X}_2 \vee \bar{X}_1X_2$			

Q=1001				
x_1x_2	00	01	10	11
$\hat{x}_1$	0011	0011	0011	0011
$\hat{x}_2$	0101	0101	0101	0101
Q	1001	1001	1001	1001
$\vec{x}_1=\hat{x}_1\oplus x_1$	0011	0011	1100	1100
$\vec{x}_2=\hat{x}_2\oplus x_2$	0101	1010	0101	1010
$\vec{L}=Q\oplus Y$	0110	1001	1001	0110
L	$X_1\bar{X}_2 \vee \bar{X}_1X_2$			

Рисунок 3.7 – Синтез дедуктивних формул для взаємно інверсних елементів: 0110 та 1001

Наступний Frame присвячений процесу синтезу дедуктивних формул для тривходового логічного елемента, заданого векторним покриттям 11001100 (рис. 3.8). Такий елемент слід розглядати як чорний ящик або RTL-рівень представлення функцій щодо його структури, яка може бути по-різному виконана при векторному завданні його поведінки. Тут інтерес становить результат транспортування списків активностей від входу до виходу цього елемента. У цьому нецікаво, які шляхи задіяні всередині конкретної реалізації логічного елемента. Тим не менш синтез дедуктивних формул для цього елемента показав, що на всіх вхідних впливах формула транспортування

активностей має одне і теж значення L= X_2. Тому можна зробити висновок, що схема має два несуттєві (надлишкові) входи з трьох.

Q=11001100								
$x_1x_2x_3$	000	001	010	011	100	101	110	111
$\hat{x}_1$	00001111	00001111	00001111	00001111	00001111	00001111	00001111	00001111
$\hat{x}_2$	00110011	00110011	00110011	00110011	00110011	00110011	00110011	00110011
$\hat{x}_3$	01010101	01010101	01010101	01010101	01010101	01010101	01010101	01010101
Q	11001100	11001100	11001100	11001100	11001100	11001100	11001100	11001100
$\vec{x}_1=\hat{x}_1\oplus x_1$	00001111	00001111	00001111	00001111	11110000	11110000	11110000	11110000
$\vec{x}_2=\hat{x}_2\oplus x_2$	00110011	00110011	11001100	11001100	00110011	00110011	11001100	11001100
$\vec{x}_3=\hat{x}_3\oplus x_3$	01010101	10101010	01010101	10101010	01010101	10101010	01010101	10101010
$\vec{L}=Q\oplus$Y	00110011	00110011	11001100	11001100	00110011	00110011	11001100	11001100
L	X_2	X_2	X_2	X_2	X_2	X_2	X_2	X_2

Рисунок 3.8 – Синтез дедуктивних формул для тривходового 11001100-елемента

Наступний Frame присвячений процесу синтезу дедуктивних формул для тривходового логічного RTL-елемента, заданого векторним покриттям 11010111 (рис. 3.9). У даному елементі всі виходи є суттєвими, тому в синтезованих дедуктивних формулах є всі вхідні активності, які транспортуються на вихід цього елемента. При цьому слід зауважити, що в останньому рядку таблиці вказані мінімізовані дедуктивні функції, отримані з ДНФ за допомогою карт Карно. Ця таблиця є результатом виконання простого паралельного алгоритму на векторних структурах даних з метою отримання дедуктивних формул для трьох і більше змінних, який може бути легко закодований будь-якою мовою програмування, включаючи HDL-мови опису апаратури.

Q=11010111								
$x_1x_2x_3$	000	001	010	011	100	101	110	111
$\hat{x}_1$	00001111	00001111	00001111	00001111	00001111	00001111	00001111	00001111
$\hat{x}_2$	00110011	00110011	00110011	00110011	00110011	00110011	00110011	00110011
$\hat{x}_3$	01010101	01010101	01010101	01010101	01010101	01010101	01010101	01010101
Q	11010111	11010111	11010111	11010111	11010111	11010111	11010111	11010111
$\vec{x}_1=\hat{x}_1\oplus x_1$	00001111	00001111	00001111	00001111	11110000	11110000	11110000	11110000
$\vec{x}_2=\hat{x}_2\oplus x_2$	00110011	00110011	11001100	11001100	00110011	00110011	11001100	11001100
$\vec{x}_3=\hat{x}_3\oplus x_3$	01010101	10101010	01010101	10101010	01010101	10101010	01010101	10101010
$\vec{L}=Q\oplus Y$	00101000	00101000	11010111	00101000	11010111	00101000	00101000	00101000
L	$\bar{X}_1X_2X_3 \vee X_1\bar{X}_2\bar{X}_3$	$\bar{X}_1X_2X_3 \vee X_1\bar{X}_2X_3$	$X_3 \vee \bar{X}_1X_2 \vee X_1\bar{X}_2$	$\bar{X}_1X_3$	$X_3 \vee \bar{X}_1X_2 \vee X_1\bar{X}_2$	$\bar{X}_1\bar{X}_2X_3 \vee X_1X_2X_3$	$\bar{X}_1X_2\bar{X}_3 \vee X_1\bar{X}_2\bar{X}_3$	$\bar{X}_1X_2X_3 \vee X_1\bar{X}_2X_3$

Рисунок 3.9 – Синтез дедуктивних формул для тривходового 11010111-елемента

Наступний Frame присвячений процесу аналізу дедуктивних формул логічної схеми управління соціальним процесом прийняття рішень щодо розгляду наукових публікацій (рис. 3.10). При цьому всі чотири xor-елементи ієрархічно фільтрують plagiarism за різними метриками. Все, що проходить на вихід схеми, є валідним з погляду новизни та відсутності плагіаризму.

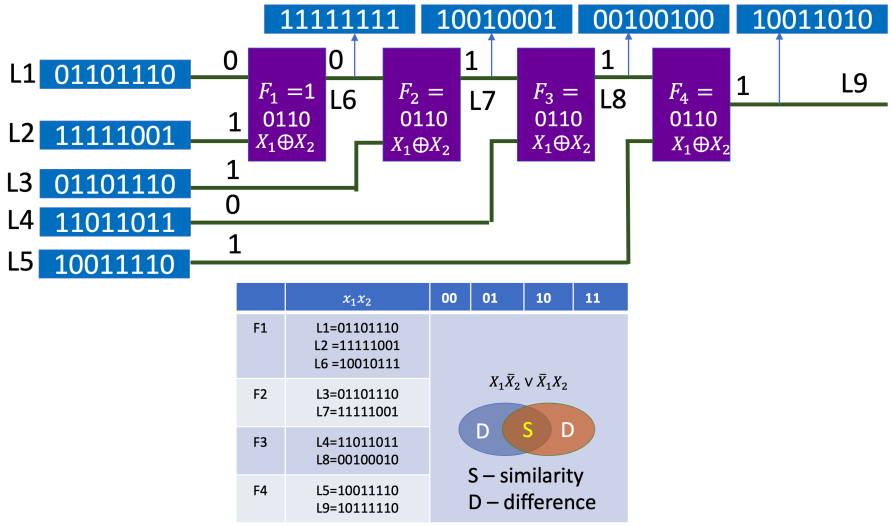


Рисунок 3.10 – Логічна 4-рівнева схема визначення новизни

Логічна схема (рис. 3.11) регулює питання проходження стартапу за такими етапами: перевірка спроможності проекту на оригінальність та капіталізацію пропозиції and-елемент, другий етап перевірка спроможності проекту на відповідність оригінальності та умовою капіталізації xor-елемент, третій етап перевірка фінансового законодавства забезпечення проекту and-

елемент, останній етап перевірки проекту щодо валідності кадрового забезпечення and-елемент.

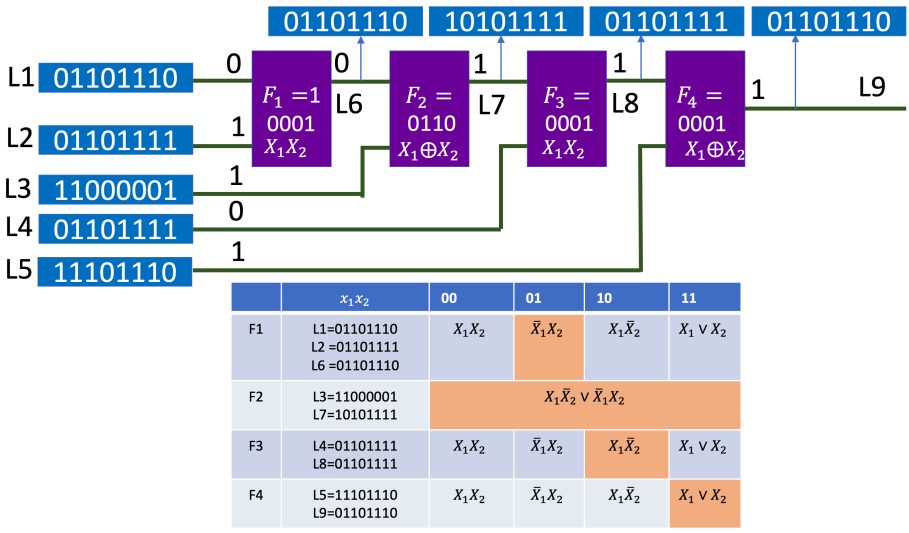


Рисунок 3.11 – Логічна 4-рівнева схема стартапу

При цьому вектори параметрів, які є у схемі, визначені на універсум–метриці примітивних атрибутів, які вимірюють усі стадії проходження проекту від специфікації до здачі проекту замовнику.

3.5 Висновки до розділу 3

Проаналізовано математичний апарат – методи дедуктивного моделювання несправностей: дедуктивно-векторний; метод векторного дедуктивного аналізу логічних структур; векторний (кубітний) метод синтезу дедуктивних функцій; синтез дедуктивних моделей основних логічних елементів.

Найпростіший спосіб побудувати механізм управління суспільством - це узагальнити досвід експертів у соціальній галузі шляхом роблення величезної кількості помилок, що призведе до формування ML-таблиці істинності. Однак для цього потрібно дуже багато часу, IT-просунутих користувачів, які

працюватимуть на федеративну ML-модель. Інший спосіб полягає у побудові точної структури управління, яка використовує досвід IT-інженерів. Тому розглянуті вище рішення спрямовані на пошук логічних схем управління суспільством, які використовують точні та перевірені механізми управління, що використовуються в сучасному комп'ютингу. Такий шлях приведе до створення моральної кібер-соціальної системи моніторингу та управління соціальними групами, державою. Те, що працює точно і без помилок у комп'ютері (логіка, арифметика, механізми моніторингу та управління), може бути застосовано в кібер-соціальній системі для прийняття правильних моральних безпомилкових рішень. Відмінності запропонованої логічної моделі управління соціальними групами полягає у використанні дедуктивних формул логічних примітивів, які формують залежність виходу схеми від вхідного впливу (режиму правління), від функції логіки (законодавства) та від списку активності (громадян) на входах кожного елемента та схеми в цілому L_y =f(x, F, L_x).

4 ДЕДУКТИВНИЙ АВТОМАТ ДЛЯ МОДЕЛЮВАННЯ НЕСПРАВНОСТЕЙ ЯК АДРЕС

Реалізується автомат моделювання несправностей логіки. Мета запропонованого дослідження – in-memory моделювання цифрових систем без синтезу.

4.1 Таблиця істинності для синтезу дедуктивного вектора

Технології верифікації використовують рівняння конволюції відносин $T \oplus F \oplus L=0$ між специфікацією, реалізацією та помилками (несправностями). Існують такі механізми вирішення проблеми: 1) Debugging – здійснює порівняння стану змінних після виконання кожного оператора; 2) механізм асерцій здійснюють порівняння стану змінних у певних точках програми та часу; 3) стандарти граничного сканування IEEE 11.49, 1500, 1687 – передбачають розбивку проекту на частини (IP-core), що дозволяє спростити завдання тестування виробу у процесі експлуатації.

Верифікація – процес визначення відповідності між специфікацією і пристроєм, що синтезується, для подальшого усунення помилок проектування за рахунок введення в логіко-часової надмірності.

Тестова верифікація – процес визначення помилок проектування на основі тесту, що синтезується (надмірність проекту) для заданого класу несправностей. Стандарти тестопридатного проектування: IEEE 11.49, 1500, 1687 дозволяють зменшити довжину тесту IP-компонентів проекту та покращити його якість за рахунок регістрів граничного сканування.

Формальна верифікація – процес досягнення якості проекту на основі використання логічно-часової надмірності (темпоральної логіки) та бібліотеки валідних схемотехнічних рішень. Тут можна використовувати темпоральну логіку SystemVerilog Assertions та продукти: Cadence Jasper Gold та Synopsys

VC Formal. Використовується мова SystemVerilog для формулювання обмежень (constraints), і розпізнавання комбінацій сигналів (functional coverage groups). Мета мови - functional-coverage driven constrained-random verification.

Дизассемблювання (відновлення) моделі – процес отримання моделі цифрового пристрою на основі хог-взаємодії тесту та заданого класу несправностей.

Моделювання несправностей – процес визначення якості синтезованого тесту на заданому класі несправності цифрового пристрою чи логічного елемента. Діагностування несправностей – процес та визначення причини та місця та типу дефектів у цифровій схемі.

Пропонується використання таблиці істинності для синтезу дедуктивного вектора (рис. 4.1), на основі логічного вектора і заданого двійкового вхідного набору, для його використання при моделюванні одиночних несправностей в цифровій схемі.

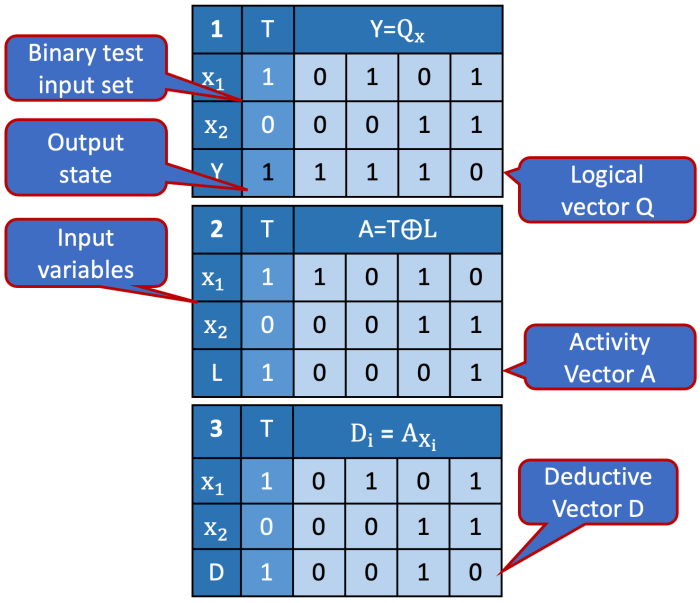


Рисунок 4.1 – Синтез дедуктивного вектора

Формула modeling: за вхідним двійковим набором, як адресою осередку логічного вектора, знаходиться стан виходу елемента, конкатенація якого з

вхідним двійковим словом формує вектор вхід-вихідних станів для хог-взаємодії з усіма стовпцями таблиці істинності з метою синтезу таблиці активності, до якої застосовується процедура впорядкування стовпців у порядку зростання двійкових адрес вхідних змінних для отримання дедуктивного вектора в таблиці впорядкованої активності.

Визначення списку вхідних несправностей логічного елемента, що перевіряються, шляхом генерації дедуктивного вектора для двійкового вхідного набору без матриці перекодування (рис. 4.2).

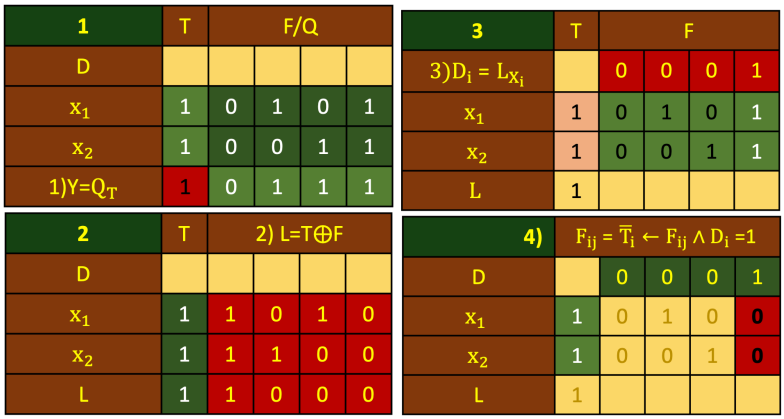


Рисунок 4.2 – Схема моделювання несправностей

Вихідні дані: вхідний двійковий набір, довжиною n: T=11 та логічний вектор Y=0111, довжиною 2^n формує 2^n вхідних наборів таблиці істинності X=00,10,01,11. Вихідні дані: таблиця істинності несправностей, що перевіряються, розмірністю $n\times 2^n$. Розмір пам'яті для зберігання структур даних визначається виразом: $M= n\times 2^n + 2 \times 2^n + n+1$, n – число змінних число змінних логічного елемента, яке формує компактні та розумні структури даних: таблиця істинності, логічний і дедуктивний вектори, вхідне двійкове слово та стан виходу.

1. На першому кроці обчислюється значення виходу Y при подачі двійкового набору 11, що розглядається як адреса біта логічного вектора $Y=Q_T=(0111)_{11}=1$.

2. Потім виконується: L=T⊕F – векторно-матрична операція отримання таблиці активності L, де стовпці, відповідні змінним x_1x_2 , розглядаються як

двійкові адреси 11, 01, 10, 00 для перекодування або переупорядкування бітів вектора L.

3. Далі виконується операція $D_{X_i}=L_i$ – переупорядкування координат L-вектора активності для отримання дедуктивного D-вектора, використовуючи двійкові стовпці таблиці істинності x_1x_2, як адреси. Після цього виконується процедура відновлення таблиці істинності. Обчислювальна складність генерації дедуктивного вектора: $2^n \times (n+1)+ 2^n$. Необхідно використовувати мову програмування (C++), яка оперує бітами інформації в таблиці істинності, з яких легко можна сконкатенувати двійкову бітову адресу двійкового вектора.

4. Формування вхідних несправностей логічного елемента, що перевіряються, інверсними значеннями вхідних сигналів змінних на одиничних координатах стовпців таблиці істинності, що покриваються (активізованих) одиничними значеннями координат дедуктивного вектора: $F_{ij} = \overline{T}_i \leftarrow F_{ij} \wedge D_i =1$. Червоний колір показує результати виконання чотирьох команд для моделювання вихідних несправностей логічного елемента. Обчислювальна складність алгоритму визначається формулою: $Q=1+2^n \times (n+1)+ 2^n+ 2^n \times n$ виконання read-write транзакцій.

4.2 Моделювання несправностей на основі таблиці істинності

Обчислення векторів вхідних несправностей функціонального елемента шляхом генерації дедуктивного вектора для двійкового вхідного набору без матриці перекодування (рис. 4.3).

Вихідні дані: вхідний двійковий набір, довжиною n: T=11 і логічний вектор Y=0111, довжиною 2^n формує 2^n вхідних наборів таблиці істинності X, у прикладі змінних x_1x_2. Вихідні дані: таблиця істинності несправностей, що перевіряються, розмірністю $n\times 2^n$. Розмір пам'яті для зберігання структур даних визначається виразом: $M= n\times 2^n + 2^n + n+1$, n – число змінних число змінних логічного елемента, яке формує компактні та розумні структури

даних: таблиця істинності, логічний (дедуктивний) вектор , вхідне двійкове слово x=11 та стану виходу Y=1.

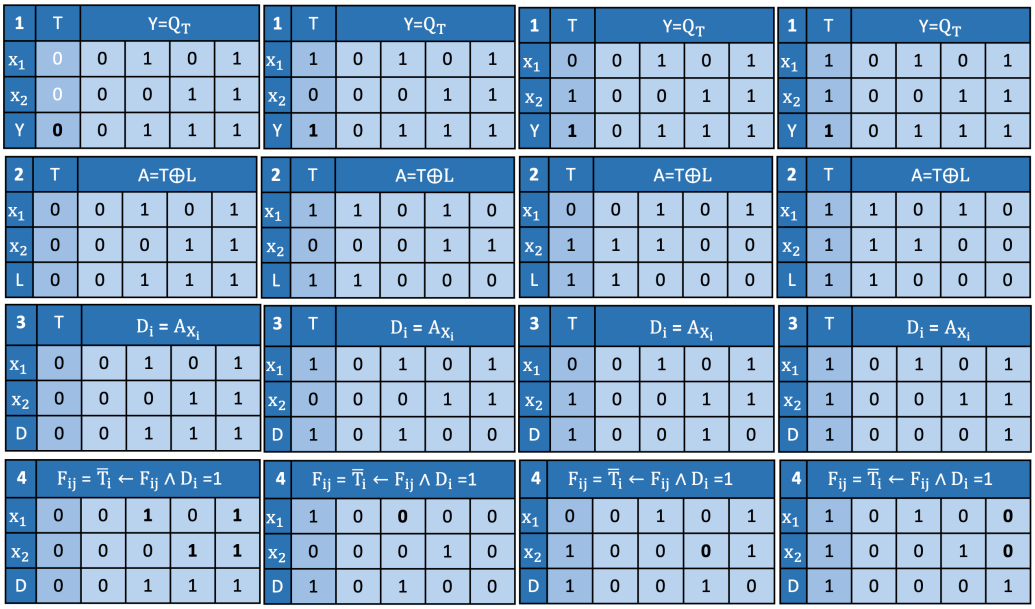


Рисунок 4.3 – Моделювання несправності на основі таблиці істинності

1) Моделювання справної поведінки елемента – обчислюється значення виходу Y під час подачі двійкового вхідного набору T, що розглядається як адреса біта логічного вектора $Y=Q_T$.

2) Векторно-матрична операція L=T для отримання таблиці активності L, де стовпці, що відповідають змінним x_1x_2 значенням розглядаються як двійкові адреси бітів дедуктивного D-вектора.

3) Упорядкування координат L-вектора за вхідними двійковими адресами x_1x_2 таблиці істинності отримання дедуктивного D вектора $D_{X_i}=L_i$. Ця операція потрібна щодо таблиці істинності до загальноприйнятому стандарту впорядкованих за зростанням двійково-десяткових адрес на основі вхідних змінних, що необхідно обробки елементів у цифровий структурі. Обчислювальна складність генерації вектора дедуктивного: $2^n \times (n+1)$.

4) Формування вхідних несправностей логічного елемента, що перевіряються, інверсними значеннями сигналів вхідних змінних x_1x_2 на одиничних координатах стовпців таблиці істинності, що покриваються

(активізованих) одиничними значеннями координат дедуктивного вектора: $F_{ij} = \overline{T}_i \leftarrow F_{ij} \wedge D_i = 1$. Обчислювальна складність алгоритму визначається наступною формулою: $Q = 1 + 2^n \times (n + 1) + 2^n + 2^n \times n$. Виконання read-write транзакцій. Пропонується використання таблиці істинності як ідеальної структури великих даних, що складається з впорядкованих по зростанню двійкових адрес осередків логічного вектора, для моделювання одиночних і кратних вхідних несправностей будь-якої функціональності на заданому двійковому вхідному тестовому наборі.

Формула моделювання: за вхідним двійковим набором, як адресою осередку логічного вектора, знаходиться стан виходу елемента, конкатенація якого з вхідних двійковим словом формує вектор вхід-вихідних станів для xor-взаємодії з усіма стовпцями таблиці істинності з метою формування таблиці активності, до якої застосовується процедура впорядкування стовпців порядок зростання двійкових адрес вхідних змінних для отримання дедуктивної таблиці істинності, де дедуктивний вектор станів виходу своїми одиничними значеннями активує одиничні координати в стовпцях таблиці несправностей, знаки яких визначаються інверсними значеннями вхідних змінних.

4.3 Моделювання несправності без упорядкування стовпців таблиці істинності

Чи можна обійтися без процедур упорядкування стовпців таблиці істинності? Так, оскільки в цьому випадку не втрачається точність та адекватність моделювання вхідних несправностей окремого логічного елемента (рис. 4.4).

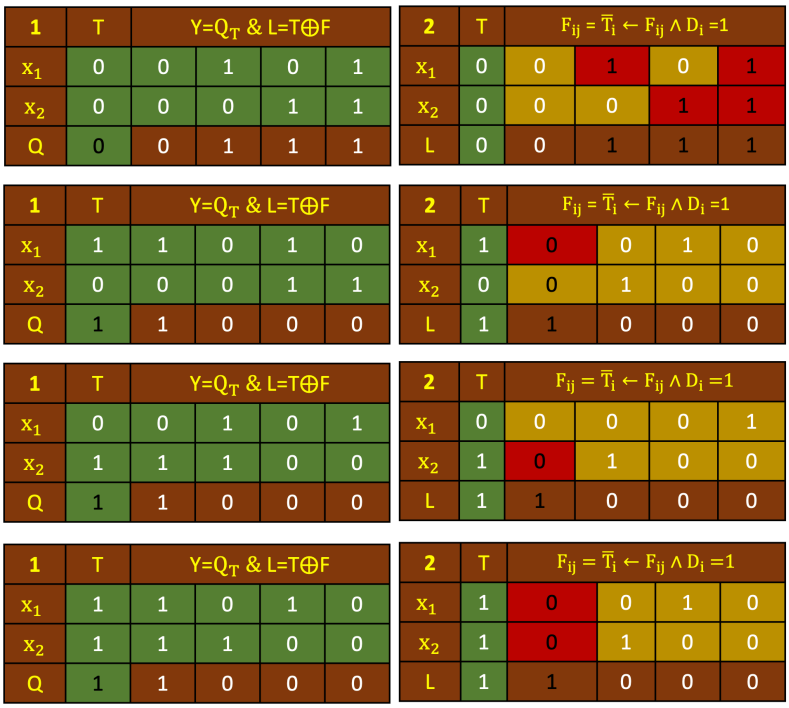


Рисунок 4.4 – Моделювання несправності без упорядкування стовпців ТІ

При цьому формула моделювання має дві процедури: за вхідним двійковим словом 11, що використовується як адреса осередку логічного вектора, знаходиться стан виходу елемента 1, який конкатенується з вхідним словом для формування вектора справного стану елемента 111, призначеного для xor-взаємодіям з усіма стовпцями таблиці істинності з метою формування невпорядковану за адресами таблиці та вектора дедукції, який своїми одиничними значеннями активізує одиничні координати відповідних стовпців таблиці істинності несправностей, знаки яких визначаються інверсними станами вхідних змінних на вхідному двійковому наборі 11.

4.4 Приклади виконання алгоритму моделювання

Приклади виконання алгоритму моделювання за пунктами з метою аналізу несправностей, що перевіряються на кожному векторі, має такий вигляд (рис. 4.5).

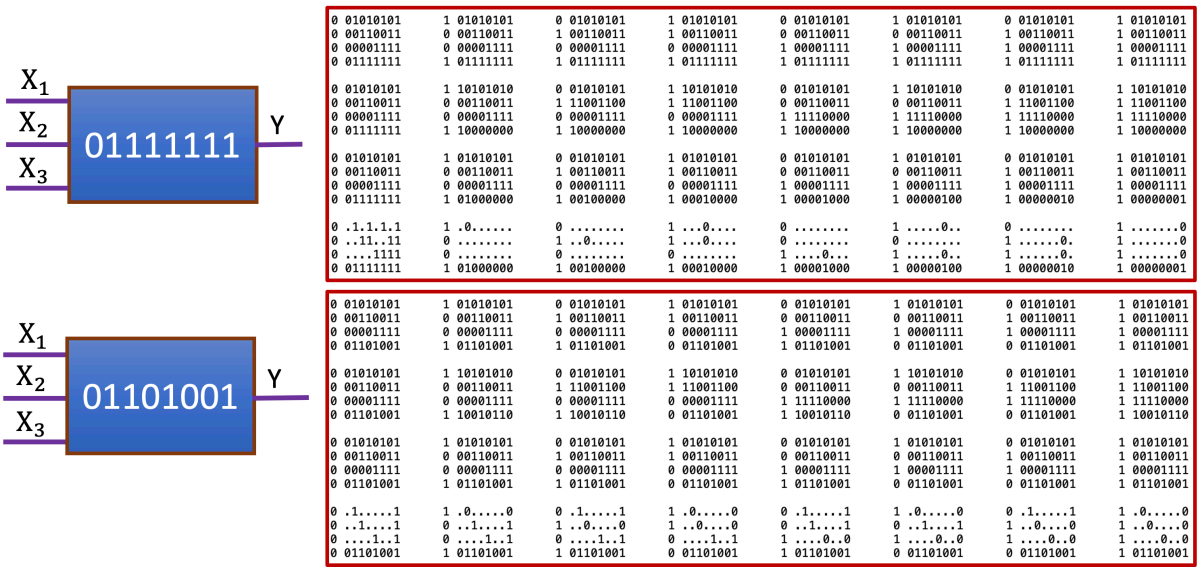


Рисунок 4.5 – Моделювання несправності 2-х входових логічних елементів

Далі на рис. 4.6 представлені аналіз функціональностей, на яких за логічним вектором, будується карта тестування всіх несправностей вхідних для логіки, представленої векторами: 00000001, 01000010.

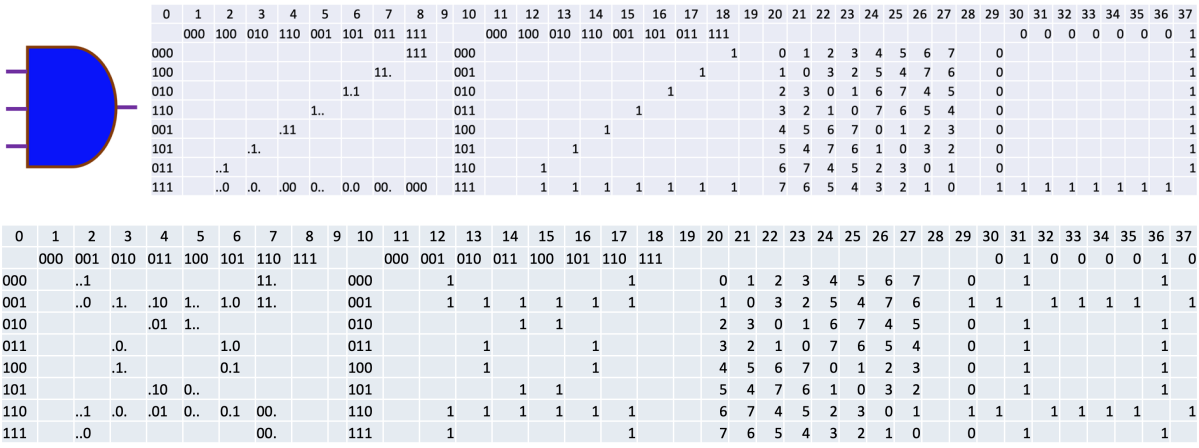


Рисунок 4.6 – Моделювання несправності 3-х входових логічних елементів

Складність функціональної логіки впливає лише на розмір виведеної інформації, необхідно вирішення практичних завдань, що з верифікації тесту та пошуком несправностей. На рис. 4.7 наведено дві схеми від чотирьох змінних, для яких побудована карта тестування.

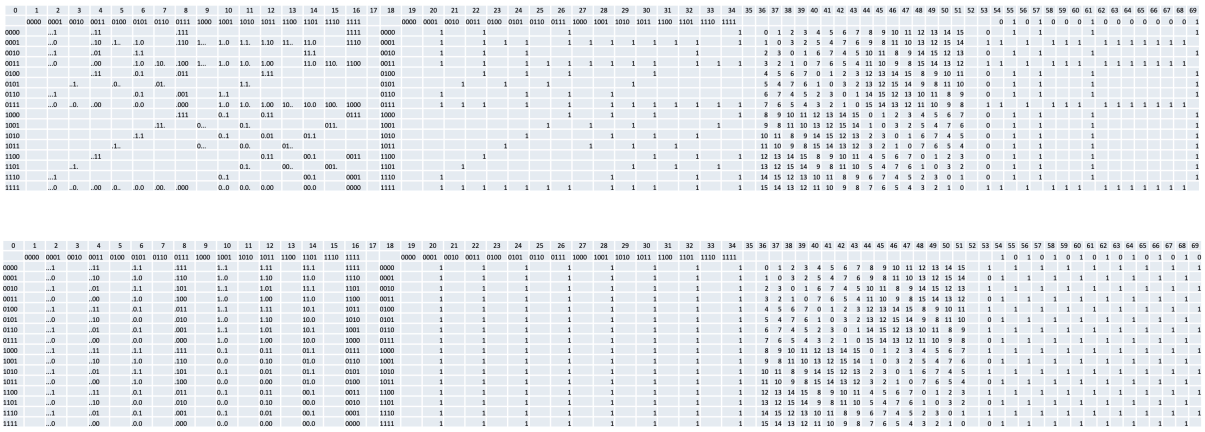


Рисунок 4.7 – Моделювання несправності 4-х входових логічних елементів

4.5 Автомат моделювання

Автомат моделювання, представлений на рис. 4.8, має на меті автоматичну перевірку складних функцій, вбудованих в IP-core SoC. Автомат використовує матрицю перекодування, яка вимагає значної пам'яті для роботи пристрою.

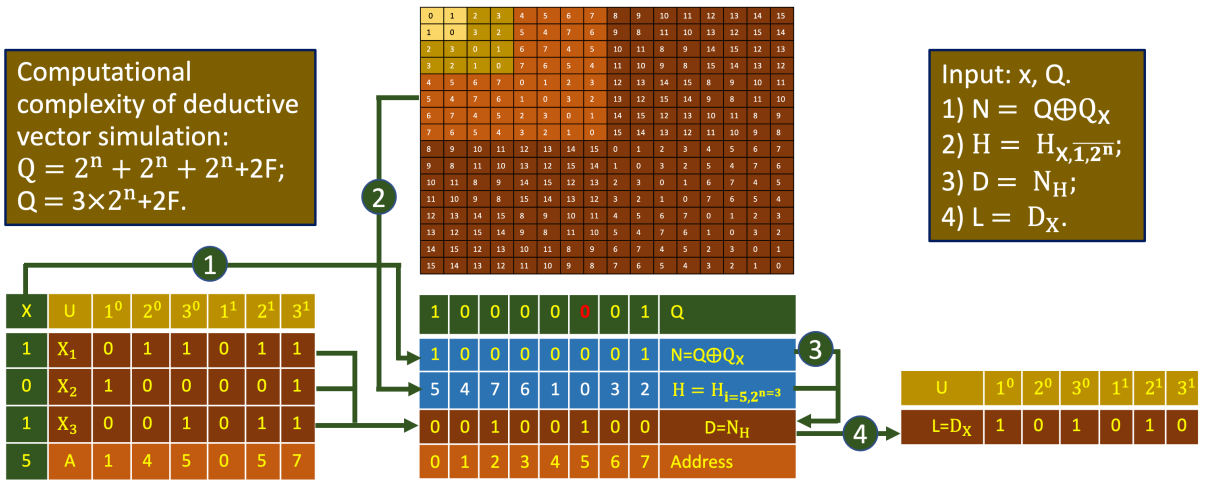


Рисунок 4.8 – Автомат моделювання несправностей логіки на основі матриці перекодування

Інтегральне рішення на основі розробленого автомата представлено на рис. 4.9. Тут показано, що на основі логічного вектора можна моделювати

будь-який вхідний набір, щоб визначати несправності, що перевіряється на цьому вхідному наборі. При цьому алгоритм створює дедуктивний вектор по матриці перекодування, яка застосовується до активного вектора, отриманого логічного вектора функціональності.

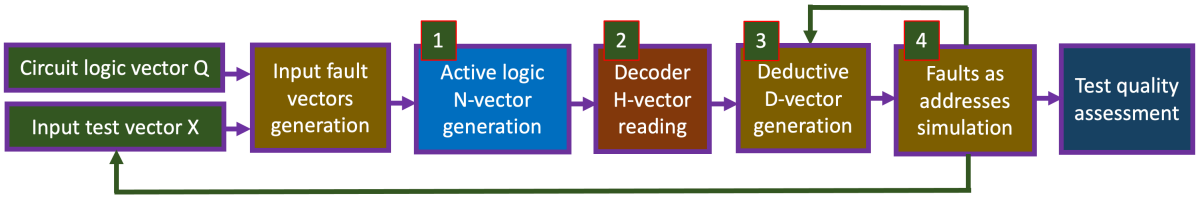


Рисунок 4.9 – Макросхема автомат моделювання несправностей логіки

Обчислювальна складність запропонованого алгоритму $Q = 2^n+2^n+2^n + 2F$ або $Q = 3\times 2^n+2F$, де F – потужність змінних ліній в схемі.

Для програмної реалізації використовується мова програмування, яка дає можливість швидко оперувати бітами в таблицях (TI) та матрицях. Серед варіантів C, C++ та Rust. Одна з найважливіших умов – це наявність доступних графічних фреймворків (бібліотек) для графічного інтерфейсу з метою створення схем і виводу інформації щодо моделювання. В результаті було обрано мову C++, оскільки вона відповідає всім описаним вище критеріям. Для C++ існує декілька графічних фреймворків, які дають можливість написати графічний інтерфейс користувача, наприклад Qt Widgets, QML, MFC, GTK. Було обрано фреймворк Qt Widgets через можливість зробити інтуїтивно зрозумілий інтерфейс будування схеми для моделювання.

4.6 Висновки до розділу 4

Таким чином, реалізовано автомат моделювання несправностей логіки на основі матриці перекодування з метою автоматичної перевірки складних функцій, вбудованих в IP-core SoC, що використовує матрицю перекодування. При цьому алгоритм створює дедуктивний вектор по матриці перекодування. Виконана програмна реалізація мовою C++ (додаток В).

ВИСНОВКИ

Мета дослідження – суттєве (20%) зниження часу верифікації цифрових схем за рахунок векторного паралельного моделювання несправностей як адрес – досягається шляхом вирішення задач:

– проаналізовано сучасні технологічні тенденції;

– виконано аналітичний огляд методів моделювання несправностей;

– вдосконалено (наукова новизна) автомат векторно-дедуктивного моделювання несправностей логіки на основі матриці перекодування з метою автоматичної перевірки складних функцій, вбудованих в IP-core SoC. Він використовує матрицю перекодування;

– розроблені структури даних (наукова новизна) та алгоритм, які формують візуальний інтерфейс введення схеми за допомогою елементів, що описуються логічними векторами або їх десятковими кодами. Економія часу (практична значимість) при введені схеми досягає кілька разів порівняно із введенням моделі за допомогою HDL.

1. Застосовано новий метод моделювання одиночних константних несправностей (вхідних, внутрішніх та вихідних) ліній схеми на основі логічних векторів функціональних елементів, які використовуються для побудови дедуктивних векторів транспортування вхідних несправностей на виходи схеми. Метод не вимагає виконання синтезу з метою приведення логічних функціональностей до певного базису елементів. Навпаки, метод швидше буде працювати, якщо макро-функціональності будуть представлені більш довгими векторами в пам'яті. Метод орієнтований на імплементацію в будь-яку пам'ять на основі виконання read-write транзакцій, що робить його вільним від потужної системи команд центрального процесора та економічним за витратами енергії та часу моделювання несправностей.

2. Несправності розглядаються як адреси для вибору відповідних біт дедуктивних векторів, що дає можливість підвищувати паралелізм обробки

векторів несправностей схеми за рахунок збільшення складності логічних елементів.

3. Модель вхідних несправностей схеми на вхідному наборі спочатку представлена квадратичною матрицею від числа лінії схеми, де діагоналі розставлені 1-значення. У процесі моделювання така матриця дозволяє легко обробляти найскладніші структурні зв'язки елементів, включаючи розгалуження, що сходяться.

4. Для обробки схеми із глобальними зворотними зв'язками необхідно вводити псевдозмінні та додаткові цикли обробки вхідних двійкових векторів для приведення результатів моделювання до стабільного вигляду.

5. Для обробки цифрових автоматів необхідно виконати попередній синтез моделі автомата з метою приведення його до векторної форми із псевдозмінними.

Результати відображено у тезах доповідях на наукових конференціях [50, 51], одна з них у науково-метричній базі Scopus (додаток Б). За період навчання в університеті опубліковано 12 наукових робіт [40-51], серед них 1 патент та дві роботи у базі Scopus.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Gartner Top 10 Strategic Technology Trends 2023 [https://www.gartner.com/en/articles/gartner-top-10-strategic-technology-trends-for-2023]

2. Bart Willemsen. Gartner Top 10 Strategic Technology Trends for 2024. https://www.gartner.com/en/articles/gartner-top-10-strategic-technology-trends-for-2024

3. Tarraf Danielle C. Control of Cyber-Physical Systems. Workshop held at Johns Hopkins University, March 2013, Springer, 2013. 378p.

4. Mohammad A. Khan, Hillol Debnath, Cristian Borcea. Balanced Content Replication in Peer-to-Peer Online Social Networks. 2016 IEEE International Conferences on Big Data and Cloud Computing, Social Computing and Networking, 2016. Pages: 274 - 283.

5. Maria R. Lee, Tsung Teng Chen. Understanding Social Computing Research. IT Professional. 2013. Volume: 15, Issue: 6 Pages: 56 - 62.

6. Jerry Higg, Varadraj Gurupur, Murat Tanik. A Transformative Software Development Framework: Reflecting the paradigm shift in social computing. 2011 Proceedings of IEEE Southeastcon. 2011 Pages: 339 - 344.

7. Cyber-Physical-Social Systems: The State of the Art and Perspectives. Jun Jason Zhang; Fei-Yue Wang; Xiao Wang; Gang Xiong; Fenghua Zhu; Yisheng Lv; Jiachen Hou; Shuangshuang Han; Yong Yuan; Qingchun Lu; Yishi Lee. IEEE Transactions on Computational Social Systems. Year: 2018, Volume: 5, Issue: 3. P. 134-144.

8. Trends Appear On The Gartner Hype Cycle For Emerging Technologies 2019: https://www.gartner.com/smarterwithgartner/5-trends-appear-on-the-gartner-hype-cycle-for-emerging-technologies-2019

9. Abramovici M. Digital System Testing and Testable Design / M. Abramovici, MA Breuer and AD Friedman.- Comp. Sc. Press.- 1998.- 652 p.

10. Vladimir Hahanov. Cyber Physical Computing for IoT-driven Services. New York. Springer. 2018. 279p.

11. Hahanov V.I. Qubit technologies for analysis and diagnosis of digital devices / V.I. Hahanov, T. Bani Amer, S.V. Chumachenko, E.I. Litvinova // Electronic Modeling.- vol. 37, no. 3.- 2015.- P. 17-40.

12. Хаханов В.И. Кубитные структуры данных вычислительных устройств / В. И. Хаханов, В. Гариби, Е. И. Литвинова, А. С. Шкиль // Электронное моделирование. - 2015. - Т. 37, № 1. - С. 76-99.

13. U. Reinsalu, J. Raik, R. Ubar and P. Ellervee, "Fast RTL Fault Simulation Using Decision Diagrams and Bitwise Set Operations," In 2011 IEEE International Symposium on Defect and Fault Tolerance in VLSI and Nanotechnology Systems, Vancouver, BC, pp. 164-170, 2011.

14. Rodrigo Fernandes de Mello, Moacir Antonelli Ponti. Machine Learning. Practical Approach on the Statistical Learning Theory. Springer, 2018. 362 p.

15. Forsyth David. Applied Machine Learning. Springer, 2019. 478 p.

16. Machine Intelligence. Essays on the Theory of Machine Learning and Artificial Intelligence, 2019. 340 c.

17. Zhengbing Hu, Yevgeniy V. Bodyanskiy, Oleksii Tyshchenko. Self-learning and Adaptive Algorithms for Business Applications: A Guide to Adaptive Neuro-fuzzy Systems for Fuzzy Clustering Under Uncertainty Conditions (Emerald Points) Paperback – June 25, 2019. 120 p.

18. C. Zhu, VCM Leung, L. Shu and ECH Ngai, "Green Internet of Things for Smart World," in IEEE Access, vol. 3, pp. 2151-2162, 2015.

19. A. Zanella, N. Bui, A. Castellani, L. Vangelista and M. Zorzi, "Internet of Things for Smart Cities," in IEEE IoT Journal, vol. 1, no. 1, pp. 22-32, Feb. 2014.

20. Frahim J. Securing the Internet of Things: A Proposed Framework / J. Frahim // Cisco White Paper.- 2015.

21. Kharchenko V. Green IT Engineering: Concepts, Models, Complex Systems Architectures / V. Kharchenko, Y. Kondratenko, J. Kacprzyk (Eds.) // In

the book series "Studies in Systems, Decision and Control" (SSDC). vol. 1. Berlin, Heidelberg: Springer International Publishing. 2017.

22. CAMPBELL, Calif., Dec. 03, 2018 (GLOBE NEWSWIRE). Wave Computing®, the Silicon Valley company that is accelerating artificial intelligence (AI) from the cloud to the edge, today announced the... [https://www.globenewswire.com/news-release/2018/12/03/1661174/0/en/Wave-Computing-Appoints-Industry-Veteran-Art-Swift-As-President-of-its-Recently-Acquired-MIPS-Licensing-Business.html]

23. Takahashi, N. Ishiura і S. Yajima, "Fault simulation for multiple faults by Boolean function manipulation," в IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, vol. 13, no. 4, pp. 531-535, April 1994, doi: 10.1109/43.275363.

24. T. Liu, T. Yu, S. Wang та S. Cai, "An Efficient Degraded Deductive Fault Simulator for Small-Delay Defects," в IEEE Access, vol. 8, pp. 204855-204862, 2020, doi: 10.1109/ACCESS.2020.3037292.

25. I. Hahanov, S. Chumachenko, I. Iemelianov, V. Hahanov, L. Larchenko and T. Daniyil, "Deductive qubit fault simulation," *2017 14th International Conference The Experience of Designing and Application of CAD Systems in Microelectronics (CADSM)*, Lviv, Ukraine, 2017, pp. 256-259, doi: 10.1109/CADSM.2017.7916129.

26. U. Reinsalu, J. Raik and R. Ubar, "Register-transfer level deductive fault simulation using decision diagrams," 2010 12th Biennial Baltic Electronics Conference, 2010, pp. 193-196, doi: 10.1109/BEC.2010.5631842.

27. I. Pomeranz and SM Reddy, "Unspecified Transition Faults: A Transition Fault Model for Speed Fault Simulation and Test Generation," в IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, vol. 27, no. 1, pp. 137-146, Jan. 2008, doi: 10.1109/TCAD.2007.907000.

28. R. Ubar, S. Devadze, J. Raik and A. Jutman, "Fast Fault Simulation for Extended Class of Faults in Scan Path Circuits," 2010 Fifth IEEE International

Symposium on Electronic Design, Test & Applications, Ho Chi Minh City , 2010, pp. 14-19, doi: 10.1109/DELTA.2010.32.

29. R. Dobai and E. Gramatova, "Deductive Fault Simulation for Asynchronous Sequential Circuits," 2009 12th Euromicro Conference on Digital System Design, Architectures, Methods and Tools, Patras, 2009, pp. 459-464, doi: 10.1109/DSD.2009.129.

30. V. Hahanov, AV Hacimahmud, E. Litvinova, S. Chumachenko та I. Hahanova, "Quantum Deductive Simulation for Logic Functions", 2018 IEEE East-West Design & Test Symposium (EWDTS), 2018, pp. 1-7, doi: 10.1109/EWDTS.2018.8524619.

31. V. Hahanov, W. Gharibi, E. Litvinova та S. Chumachenko, "Qubit-driven Fault Simulation," 2019 IEEE Latin American Test Symposium (LATS), 2019, pp. 1-7, doi: 10.1109/LATW.2019.8704583.

32. W. Gharibi, D. Devadze, V. Hahanov, E. Litvinova та I. Hahanov, «Qubit Test Synthesis Processor for SoC Logic," в 2019 IEEE Веб-Вест Design & Test Symposium (EWDTS), Batumi, Georgia, 2019 pp.1-5 doi: 10.1109/EWDTS.2019.8884476.

33. V. Hahanov et al., "Vector-Qubit models for SoC Logic-Structure Testing and Fault Simulation," 2021 IEEE 16th International Conference on Experience of Designing and Application of CAD Systems (CADSM), 2021, pp. 24-28, doi: 10.1109/CADSM52681.2021.9385266.

34. VI Hahanov, SM Hyduke, W. Gharibi, EI Litvinova, SV Chumachenko та IV Hahanova, "Quantum Models and Method for Analysis and Testing Computing Systems," 2014 11th International Conference on Information Technology: New Generations, Las Vegas, NV, 2014, pp. 430-434, doi: 10.1109/ITNG.2014.125.

35. M. Karavay, V. Hahanov, E. Litvinova, H. Khakhanova та I. Hahanova, "Qubit Fault Detection in SoC Logic," pp. 1-7, doi: 10.1109/EWDTS.2019.8884475.

36. M. Dumitrescu, T. Munteanu, D. Floricau та AP Ulmeanu, "A complex fault-tolerant power system simulation," 2005 2nd International Conference on Electrical and Electronics Engineering, 2005, pp. 267-272, doi: 10.1109/ICEEE.2005.1529624.

37. S. Moazzeni, A. Emami і S. Poormozaffari, "An Optimized Simulation-Based Fault Injection and Test Vector Generation Using VHDL to Calculate Fault Coverage," 2009 10th International Workshop on Microprocessor Test and Verification, 2009, pp. 55-60, doi: 10.1109/MTV.2009.22.

38. S. Teshima, N. Chujo, N. Sano, H. Nagase і M. Takigawa, "Accelerated fault simulation by propagating disjoint fault-sets," [1988] The Eighteenth International Symposium on Fault-Tolerant Computing. Digest of Papers, 1988, pp. 116-121, doi: 10.1109/FTCS.1988.5308.

39. M. Srivastava, SK Goyal, A. Saraswat and G. Gangil, "Simulation Models for Different Power System Faults," 2020 IEEE International Conference on Advances and Developments in Electrical and Electronics Engineering (ICADEE), 2020, pp. 1-6, doi: 10.1109/ICADEE51157.2020.9368915.

40. Филиппенко И.В. Встроенная система регулирования температуры в «Умном доме» [Текст] / И.В. Филиппенко, С.Э. Кондрюков, Г.К. Кулак // Радиоэлектроника и информатика. 2017. № 3(78) С. 23-27.

41. Шкиль А.С. Шаблоны автоматного программирования для проектирования устройств логического управления на основе конечных автоматов [Текст] / А.С. Шкиль, Г.К. Кулак // Тези доповідей шостої міжнародної науково-технічної конференції Черкаси – Баку– Бельско-Бяла – Харків. – 14-16 листопада 2018р. с. 39-40.

42. Патент України на корисну модель № 132857, МПК (2018.10) G02C 5/00, G02C 5/02(2006.10) публ.11.03.2019, бюл. №5 Оправа окулярів для контролю безпечної відстані для очей при читанні, Кулак Ельвіра Миколаївна, Кулак Георгій Костянтинович.

43. Кулак Г.К. Кубическая резонансная решетка металлических сфер в магнитодиэлектрической среде [Текст] / Г.К. Кулак // Тези доповіді 23-го

Міжнародного молодіжного форуму «Радіоелектроніка та молодь у 21 столітті» Зб. матеріалів форуму Т.3 – Харків.: ХНУРЕ – 16-18 квітня 2019. с. 10-11.

44. Шкіль А.С. Аналіз коректності графових моделей керуючих автоматів для автоматизованого синтезу [Текст] / А.С. Шкіль, Г.К. Кулак // Тези доповідей семої міжнародної науково-технічної конференції «Проблеми інформатизації» Черкаси – Харків – Баку – Бельско-Бяла. – 13-15 листопада 2019р. с. 31.

45. Шкіль А.С. Мікроконтролерна система управління на основі богатозадачності. [Текст] / А.С. Шкіль, І.В. Філіппенко, Г.К. Кулак, Д.О. Семенцов // Тези стендових доповідей та виступів учасників 32 міжнародної науково-практичної конференції «Інформаційно-керуючі системи на залізничному транспорті», Харьков. - №4 (приложение), – 24-25 жовтня 2019. С. 75-76.

46. Філіппенко І.В., Огляд графічних бібліотек для вбудованих платформ [Текст] / І.В. Филиппенко, В.Р. Корнієнко, Г.К. Кулак // Радиоэлектроника и информатика. 2020. № 1(1) С. 51-59.

47. Кулак Е.М. Метод пошуку помилок в умовах переходів графових моделей керуючих автоматів [Текст] / Е.М. Кулак, О.О. Погудін, Г.К. Кулак // Тези доповідей восьмої міжнародної науково-технічної конференції «Проблеми інформатизації» Черкаси – Харків – Баку – Бельско-Бяла. – 26-27 листопада 2020р. том 1, с. 53.

48. Філіппенко І.В. Використання мікроконтролерів при створенні нейронних мереж [Текст] / І.В. Філіппенко, В.Р. Адамович, Г.К. Кулак // Тези доповідей одинадцятої міжнародної науково-технічної конференції «Сучасні напрями розвитку інформаційно-комунікаційних технологій та засобів управління» Баку – Харків – Київ – Жиліна. – 8-9 квітня 2021р. с. 106-107.

49. Shkil A. Assertion Based Design of Timed Finite State Machine [Text] / A. Shkil, G. Kulak, A. Miroshnyk, K. Pshenychnyi // Proceedings of 2021 IEEE

East-West Design & Test Symposium (EWDTS'21), September, 10-13, Batumi, Georgia, 2021. – P.291-294 (Scopus)

50. W. Gharibi, S. Chumachenko, M. Miroshnyk, M. Abashidze, E. Litvinova, G. Kulak, V. Hahanov, A. Mishchenko, "In-Memory Fault as Address Simulation," *2023 IEEE East-West Design & Test Symposium (EWDTS)*, Batumi, Georgia, 2023, pp. 1-7, doi: 10.1109/EWDTS59469.2023.10297038 (Scopus).

51. Кулак Г.К. Моделі логічних процесорів social-комп'ютингу // Матеріали Міжнародного радіоелектронного форуму «Радіоелектроніка та молодь у XX столітті». 15.05.2023. Харків, Україна. С. 119-120.