ДОДАТОК А

Графічний матеріал кваліфікаційної роботи

NURE
Харківський національний університет радіоелектроніки

Магістерська кваліфікаційна робота

Методи забезпечення самовідновлення програмного забезпечення хмарних інформаційних систем

Здобувач гр. СПзм-23-1 Галушка О.І
Керівник ст. викл. каф. ЕОМ Бугрій А.М.

Харків, 2025

Актуальність дослідження методів забезпечення самовідновлення програмного забезпечення

Хмарні обчислення — ключовий напрям сучасної IT-інфраструктури. Забезпечення безперервності та стійкості хмарних сервісів є критично важливим, зокрема для e-commerce, банківських систем, охорони здоров'я. Робота торкається викликів самовідновлення — технології, що є у фокусі провідних розробників (Amazon, Google Cloud, IBM). Використання штучного інтелекту (KNN, нечітка логіка) у fault-tolerance — це сучасний, перспективний підхід.

Google Cloud IBM Cloud aws Azure

Недоліки існуючих методів:

- **Низька адаптивність**: більшість підходів не враховують динамічний стан ресурсів і завдань.
- **Фіксовані стратегії відновлення** (тільки реплікація або контрольні точки), незалежно від умов.
- **Відсутність механізмів самонавчання** та аналізу історичних даних про збої.
- **Неврахування пріоритетів користувача** та можливостей хмарної інфраструктури.
- **Високі накладні витрати** при нераціональному використанні реплікації.

2

Мета та задачі роботи

Метою роботи є розроблення адаптивного методу забезпечення відмовостійкості у хмарному середовищі з використанням самовідновлення, який б динамічно обирал найбільш ефективний метод fault-tolerance (реплікація або контрольні точки) на основі аналізу ризику відмов, пріоритету завдання та доступних ресурсів.

Завдання дослідження

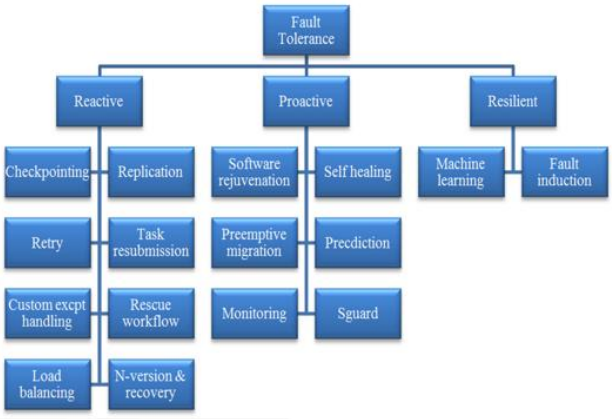
1.Проаналізувати сучасні підходи до fault-tolerance у хмарних системах.
2.Сформулювати критерії вибору між методами реплікації та контрольних точок.
3.Запровадити метод класифікації рівня ризику машин з використанням KNN.
4.Розробити систему нечіткого висновку FTMSFIS для прийняття рішень.
5.Провести моделювання та оцінку ефективності системи в умовах хмарного середовища.

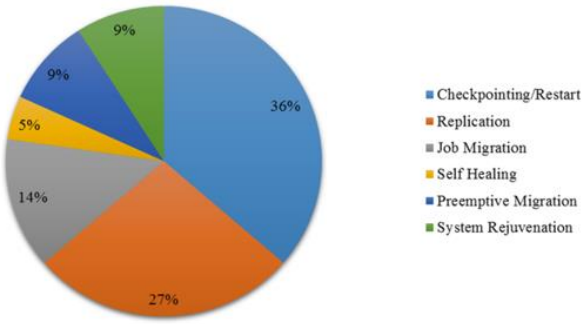
Об'єкт дослідження: Процеси забезпечення відмовостійкості та самовідновлення у хмарних обчислювальних системах.

Предмет дослідження: Методи й моделі адаптивного вибору механізмів fault-tolerance (реплікація або контрольні точки) на основі класифікації рівня ризику фізичних машин, нечіткого логічного висновку та пріоритетів користувацьких завдань у хмарному середовищі.

3

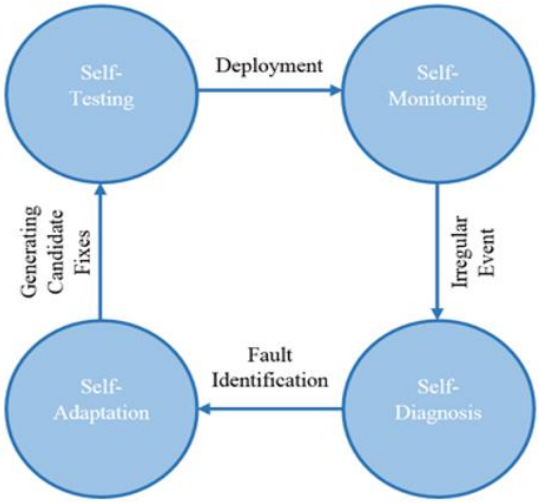
Аналіз методів забезпечення відмовостійкості





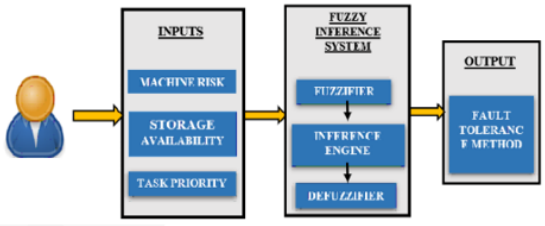
4

Архітектура системи забезпечення відмовостійкості з використанням методів самовідновлення



5

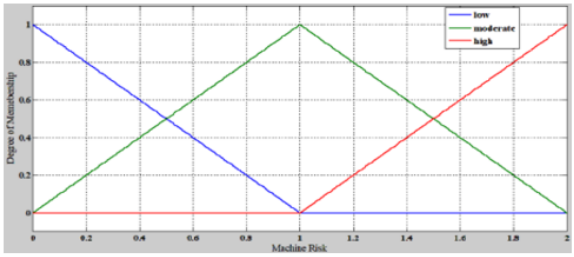
Запропонована модель забезпечення відмовостійкості та характеристики вхідних даних для FAZZY - логіки



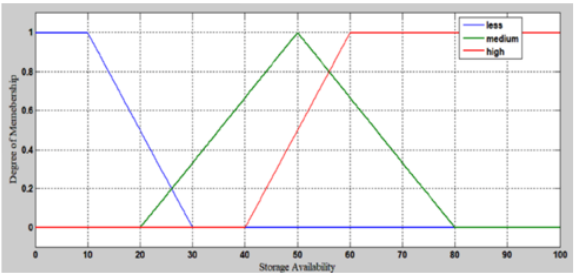
Змінні	Тип даних	Рівень ризику/назва методу	Кінцеві точки інтервалу	
			Діапазон	Точки
Ризик	Вхідні	Low	(0-2)	(0,0,1)
		Medium		(0,1,2)
		High		(0,2,2)
Доступність сховища даних	Вхідні	Low	(0-100)	(0,0,15,30)
		Medium		(15,50,85)
		High		(30,60,100)
Пріоритет завдання	Вхідні	Low	(0-2)	(0,0,5)
		Medium		(3,6,9)
		High		(6,8,10)
Метод самовідновлення	Вихідні	Контрольні точки	(0-90)	(0,35,60)
		Реплікація		(35, 50, 90)

6

Нечіткі правила (1)



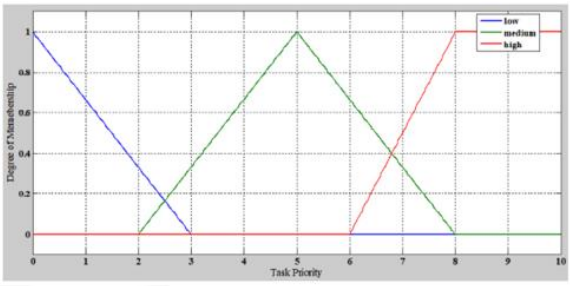
Нечітке правило машинного ризику (MR)



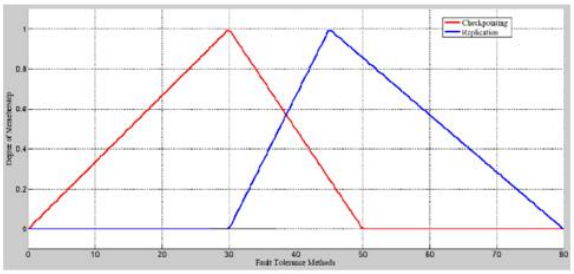
Нечітке правило для доступності сховища даних (SA)

7

Нечіткі правила (2)



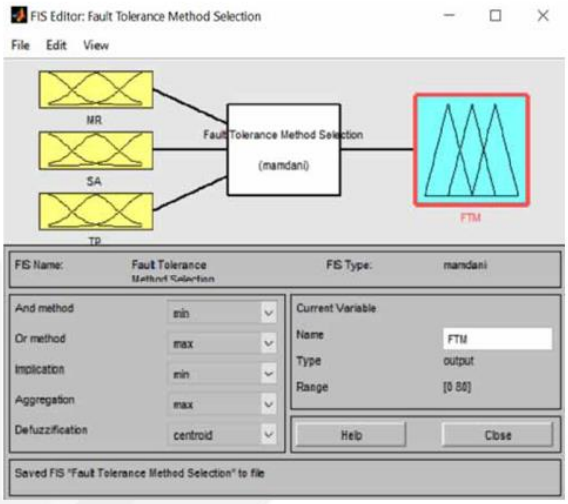
Нечітке правило для пріоритету завдання (TP)

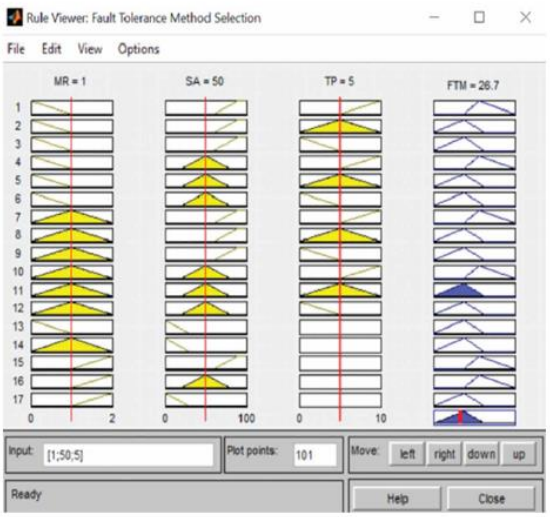


Нечітке правило для вибору метода самовідновлення (FTM)

8

Використання MATLAB для моделювання нечітких правил





9

Реалізовані класи для забезпечення самовідновлення

```python
class AutonomicComputing:
    def __init__(self):
        self.system_state = "optimal"

    def monitor_system(self):
        # Simulate system monitoring
        return "System stable"

    def detect_issue(self):
        # Detect potential issues
        if self.system_state != "optimal":
            return True
        return False

    def heal(self):
        # Self-healing process
        if self.detect_issue():
            self.system_state = "optimal"
            return "Issue resolved"
        return "No issues detected"

# Example usage
ac = AutonomicComputing()
ac.monitor_system()
ac.heal()
```

Об'явлення класу автоматичних обчислень

```python
class ResilienceEngineering:
    def __init__(self):
        self.service_availability = 99.9  # percentage

    def detect_disruption(self):
        # Simulate detection of a disruption
        return "Disruption detected"

    def recover(self):
        # Self-healing process
        self.service_availability = 100
        return "Service availability restored"

# Example usage
re = ResilienceEngineering()
re.detect_disruption()
re.recover()
```

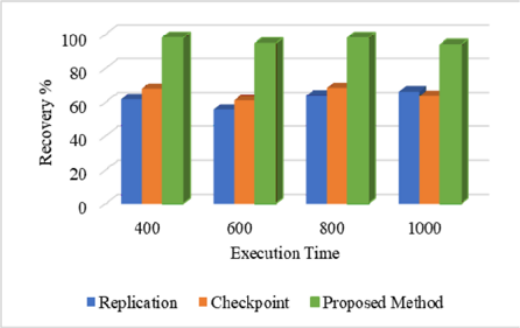
Об'явлення класу забезпечення самовідновлення

```python
import random

class SelfHealingSystem:
    def __init__(self):
        self.downtime = 0
        self.performance = 100  # percentage

    def monitor_performance(self):
        # Simulate random performance drops
        self.performance -= random.randint(0, 20)
        return self.performance

    def automated_heal(self):
        if self.performance < 80:
            self.downtime += 1
            self.performance = 100  # Restore performance
        return f"Downtime: {self.downtime}, Performance: {self.performance}"

# Example usage
shs = SelfHealingSystem()
shs.monitor_performance()
shs.automated_heal()
```

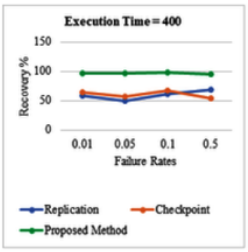
Клас ініціалізації системи самовідновлення

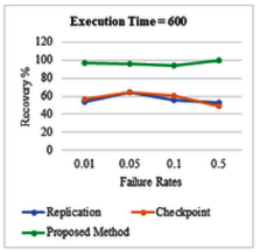
10

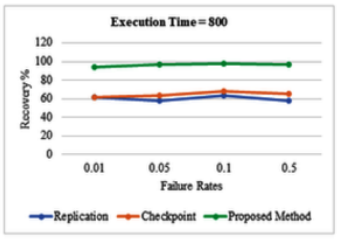
Результати експериментів



Успішне відновлення віртуальних машин







Успішне відновлення проти частоти відмов

11

ВИСНОВКИ

В процесі роботи реалізовано адаптивній метод забезпечення відмовостійкості у хмарному середовищі з використанням самовідновлення, який динамічно обирає найбільш ефективний метод fault-tolerance (реплікація або контрольні точки) на основі аналізу ризику відмов, пріоритету завдання та доступних ресурсів.

Вирішені наступні задачі дослідження.

1. Проаналізовано сучасні підходи до fault-tolerance у хмарних системах.
2. Сформульовано критерії вибору між методами реплікації та контрольних точок.
3. Запроваджено метод класифікації рівня ризику машин.
4. Розроблено систему нечіткого висновку для прийняття рішень.
5. Проведено моделювання та оцінку ефективності системи в умовах хмарного середовища.

Публікація:

1. Волк М.О., Бугрій А.М., Бітюкова Є.В. та інші. Розподілене моделювання гетерогенних систем інтернету речей. Вісник Херсонського національного технічного університету. Том 2 No 1(92). 2025. - С. 45-50. DOI: https://doi.org/10.35546/kntu2078-4481.2025.1.2.6

12