Акулич Кирилл,853404,Лабораторная работа 4

Для моделирования сетей петри был использован язык Питон.
Пару слов о реализации.

1. Для представления сетей петри во время их работы используется их матричное представление. Ведь в таком случае переходы можно осуществлять переходы и проверять некоторые св-ва сети используя элементарные матричные операции(сложение векторов и покомпонентное сравнение)

4.2.2. Матричные уравнения

Второй подход к анализу сетей Петри основан на матричном представлении сетей Петри. Альтернативным по отношению к определению сети Петри в виде (P, T, I, O) является определение двух матриц D^- и D^+, представляющих входную и выходную функции. (Они эквивалентны функциям F и B определения Хэка сетей Петри, см. разд. 2.6.) Каждая матрица имеет m строк (по одной на переход) и n столбцов (по одному на позицию). Определим $D^-[j, i] = \#(p_i, I(t_j))$, а $D^+[j, i] = \#(p_i, O(t_j))$. D^- определяет входы в переходы, D^+ — выходы.

Матричная форма определения сети Петри (P, T, D^-, D^+) эквивалентна стандартной форме, используемой нами, но позволяет дать определения в терминах векторов и матриц. Пусть $e[j]$ — m-вектор, содержащий нули везде, за исключением j-й компоненты. Переход t_j представляется m-вектором $e[j]$[2].

Теперь переход t_j в маркировке μ разрешен, если $\mu \geqslant e[j] \cdot D^-$, а результат запуска перехода t_j в маркировке μ записывается как

$$\delta(\mu, t_j) = \mu - e[j] \cdot D^- + e[j] \cdot D^+ =$$
$$= \mu + e[j] \cdot (-D^- + D^+) = \mu + e[j] \cdot D,$$

где $D = D^+ - D^-$ — составная матрица изменений.

2. Для анализа сетей используется графовое представление сетей петри.(Анализ с помощью матриц не очень удобен,т.к. пока нету в python возможности решать квадратные системы уравнений).
3. Для проверки правильности работы используем пакет для анализа сетей из MathLab

Анализ первой сети

определим сеть с помощью матриц и зададим начальное состояние сети(mu_0)

```
D_input = np.array([
   [0,0,0,1,0],
   [1,1,0,0,0],
   [0,0,1,0,0],
   [0,0,0,1,0]])
D_output = np.array([
   [1,0,0,0,0],
```

```
   [0,0,1,1,0],
   [0,1,0,0,0],
   [0,0,0,0,1]])
D =   D_output - D_input
mu_0 = (0,0,1,1,0)
```

Схема из лабораторной

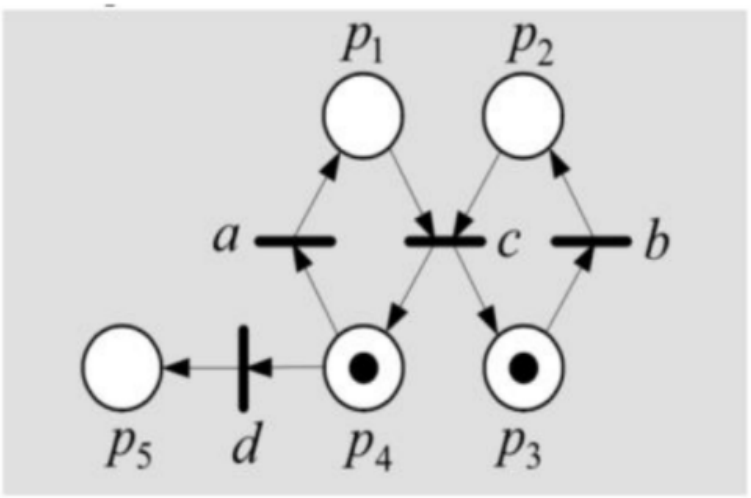


Схема из МатЛаб

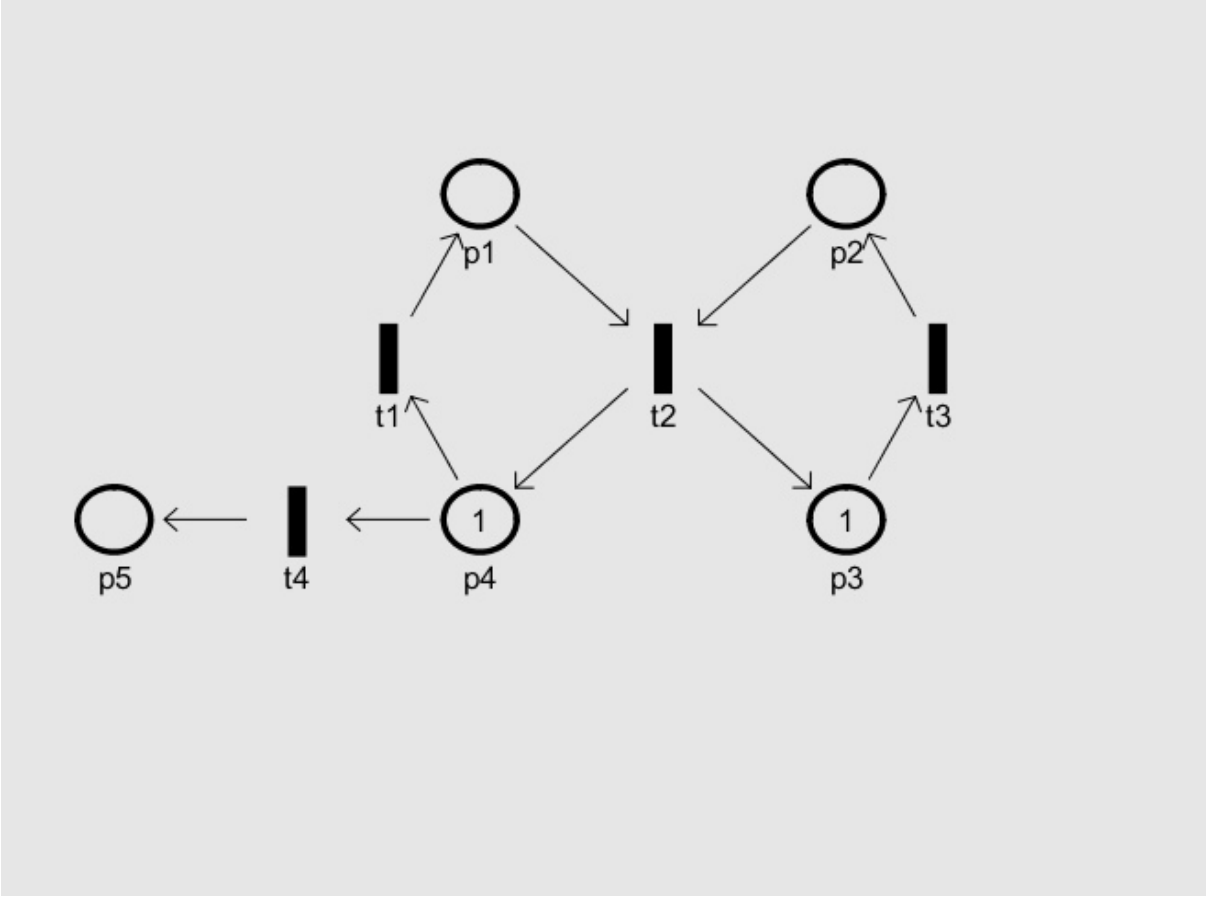
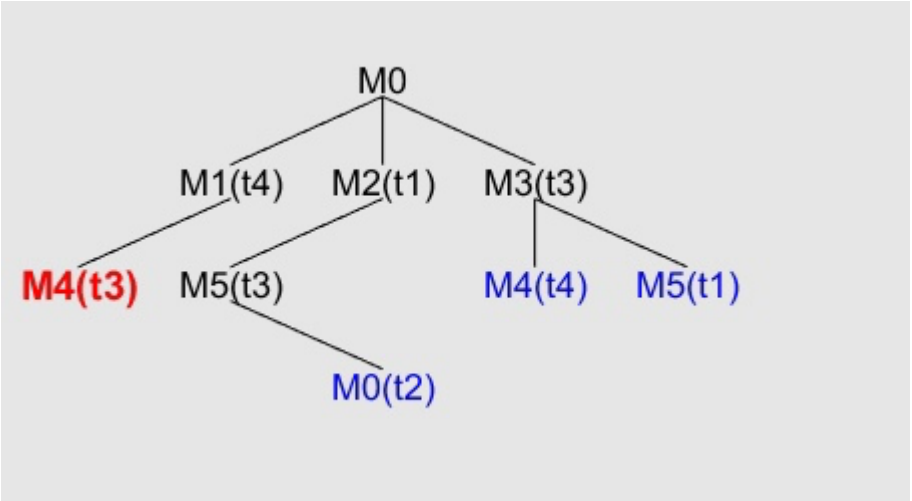


Диаграмма маркировок(Матлаб):



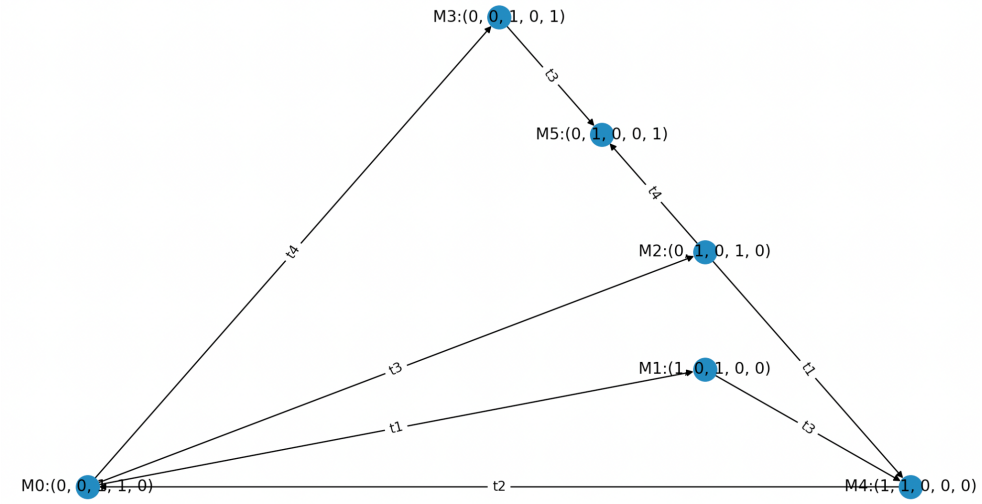
Coverability Tree - Text Mode

From	Fired	To	From	Fired	To	From	Fired	To	From	Fired	To	From	Fired	To	From	Fired	To
M0	t1	M1	**M0**	**t2**	**M2**	M0	t4	M3	**M1**	**t4**	**M4**	M2	t4	M5	**M3**	**t1**	**M4**
M3	t2	M5	**M5**	**t3**	**M0**												

M[p1,p2,p3,p4,p5]

M0 = [0,0,1,1,0]	M1 = [0,0,1,0,1]	M2 = [1,0,1,0,0]
M3 = [0,1,0,1,0]	M4 = [0,1,0,0,1]	M5 = [1,1,0,0,0]

Диаграмма маркировок(программа)



Свойства сети

- Тупиковая(неживая). Существует состояние (0,1,0,0,1), из которого нет возможности выбраться

```
def isLive(G):
    for node in G.nodes:
        if len(list(G.neighbors(node))) == 0:
            return False
    else:
        return True
```

- Сеть является ограниченной, k = 1.

```
def Bounded(G,inifinity_nodes):
```

```
    if len(inifinity_nodes)!=0:
        return np.inf
    nodes = G.nodes
    return max([max(node) for node in nodes])
```

- Является безопасной, т.к ограничена и k=1
- Является консервативной(сумма в состояниях постоянна)

```
def Conservative(G):
    sums = [sum(node) for node in G.nodes]
    return len(set(sums)) == 1
```

- Срабатывание переходов одновременно.Возможно, например из состояния (0, 1, 0, 1, 0) с помощью переходов t1 t3

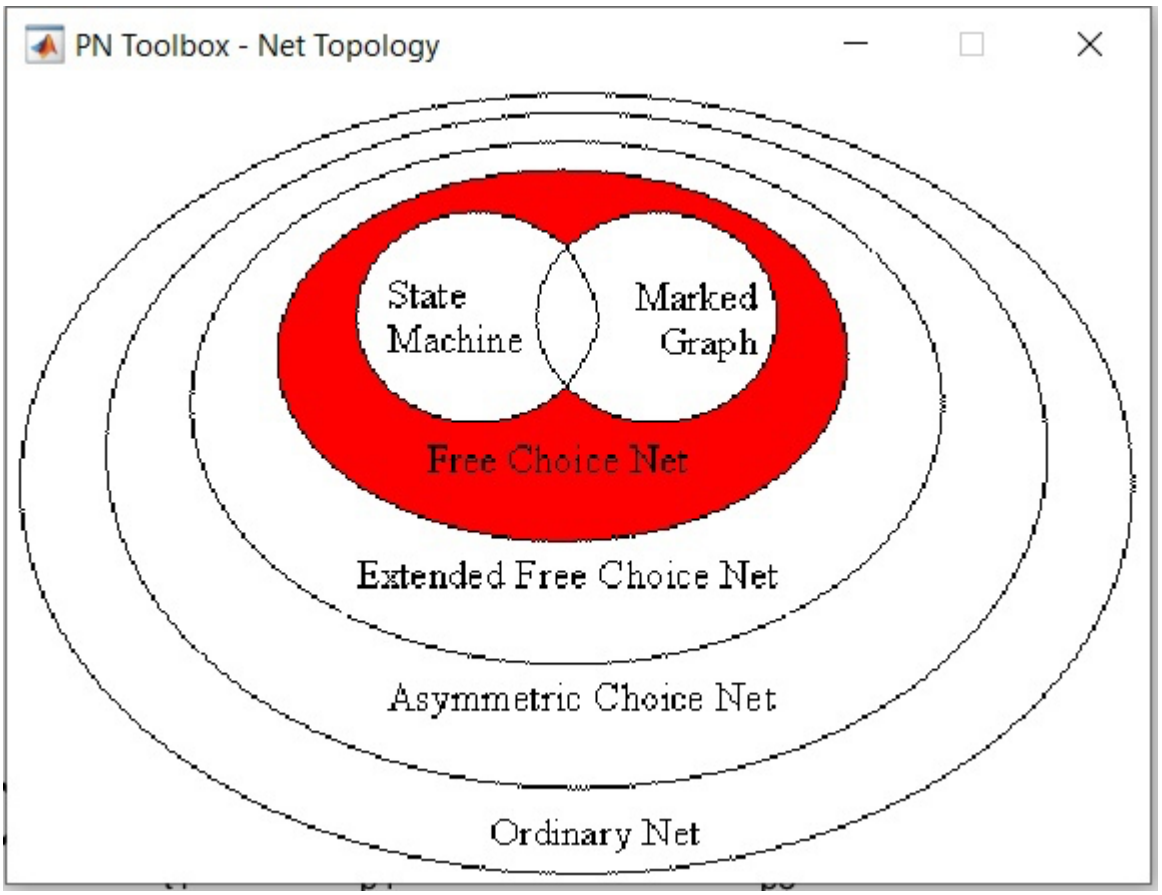
```
def ParallelTransavtions(G,D_input):
    for node in G.nodes:
        neighbors = G.neighbors(node)

        if len(list(neighbors)) < 2:
            continue
        for i in range(np.shape(D_input)[0]):
            new_state = np.array(node) - D_input[i]
            if np.any(new_state<0):
                continue
            for j in range(i+1,np.shape(D_input)[0]):
                if np.all((new_state - D[j])>=0):
                    print(f'Параллельное срабатывание вохможно: Из состояния {node}
с помощью переходов t{i+1} t{j+1}')
                    return
    print('Параллельное срабатывание невозможно')
```

- Сеть является сетью свободного выбора.Т.к для любых tiεT и piεI(ti) позиция pi является либо единственной входной позицией перехода ti , либо этот переход имеет единственную входную позицию.

```
def if_freechoice_net(D_input,D_output):
    for i in range(np.shape(D_input)[0]):
        for j in range(np.shape(D_output)[1]):
            if D_input[i,j] >1 or D_output[i,j] > 1:
                return False
    return True
```

Проверка типа сета в матлаб



Метод для определения достижимости вершин в графе

```
def can_be_reachable(G,source,target):
    return nx.has_path(G,source,target)
```

Анализ второй сети

Структура сети

```
D_input = np.array([
    [0,0,0,1,0],
    [1,1,0,0,0],
    [0,0,1,0,0],
    [0,0,0,1,0]])
D_output = np.array([
    [1,0,0,0,0],
    [0,0,1,1,0],
    [0,1,0,0,0],
    [0,0,0,0,1]])
D =    D_output - D_input
mu_0 = (0,0,1,1,0)
```

Схема из лабораторной

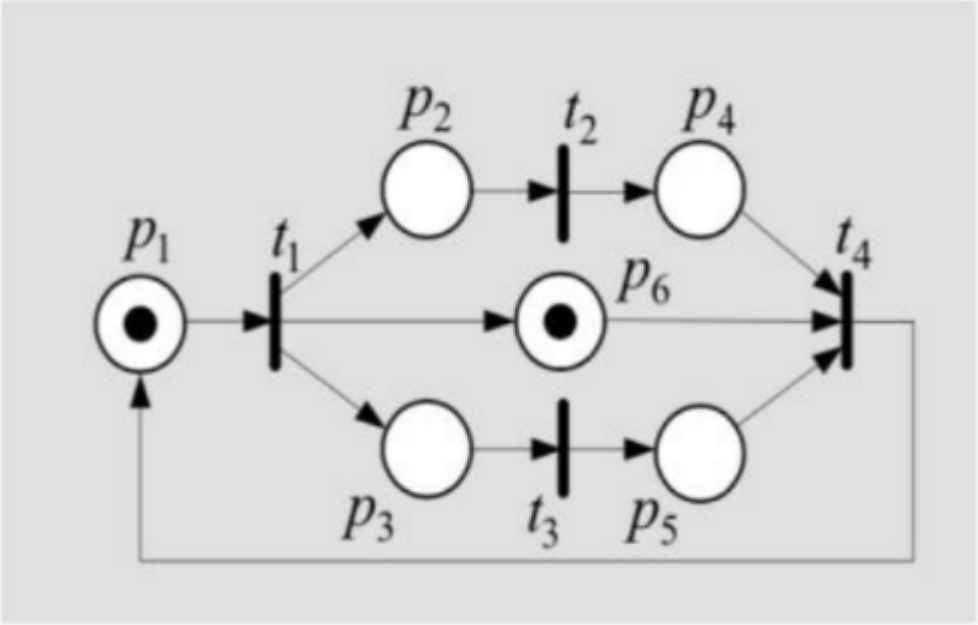


Схема из МатЛаб

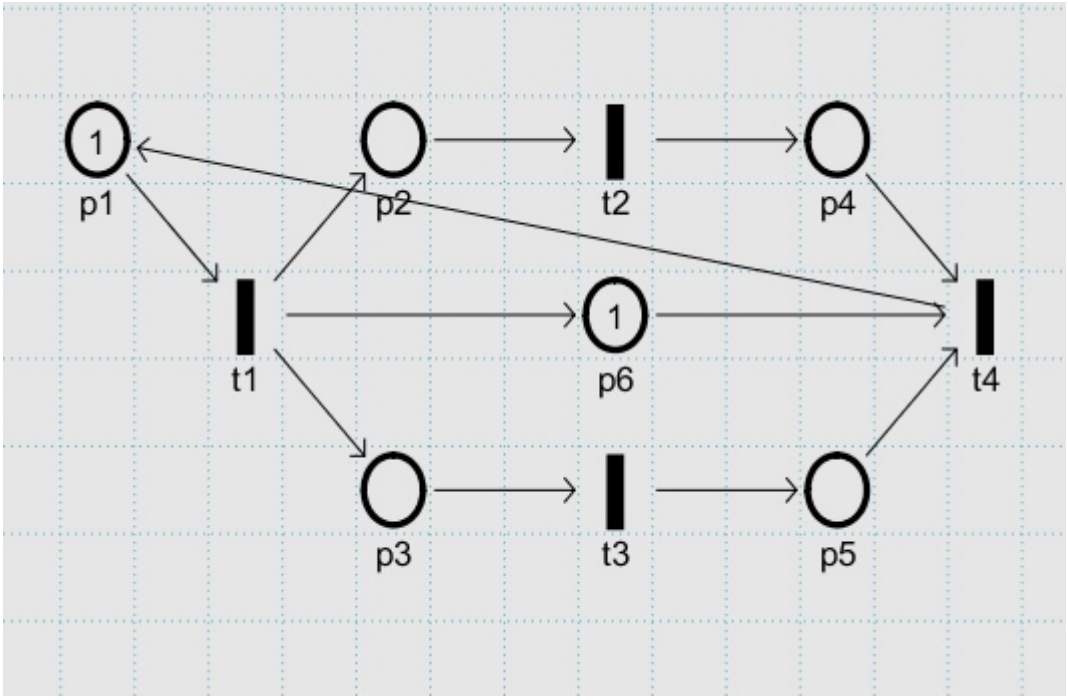
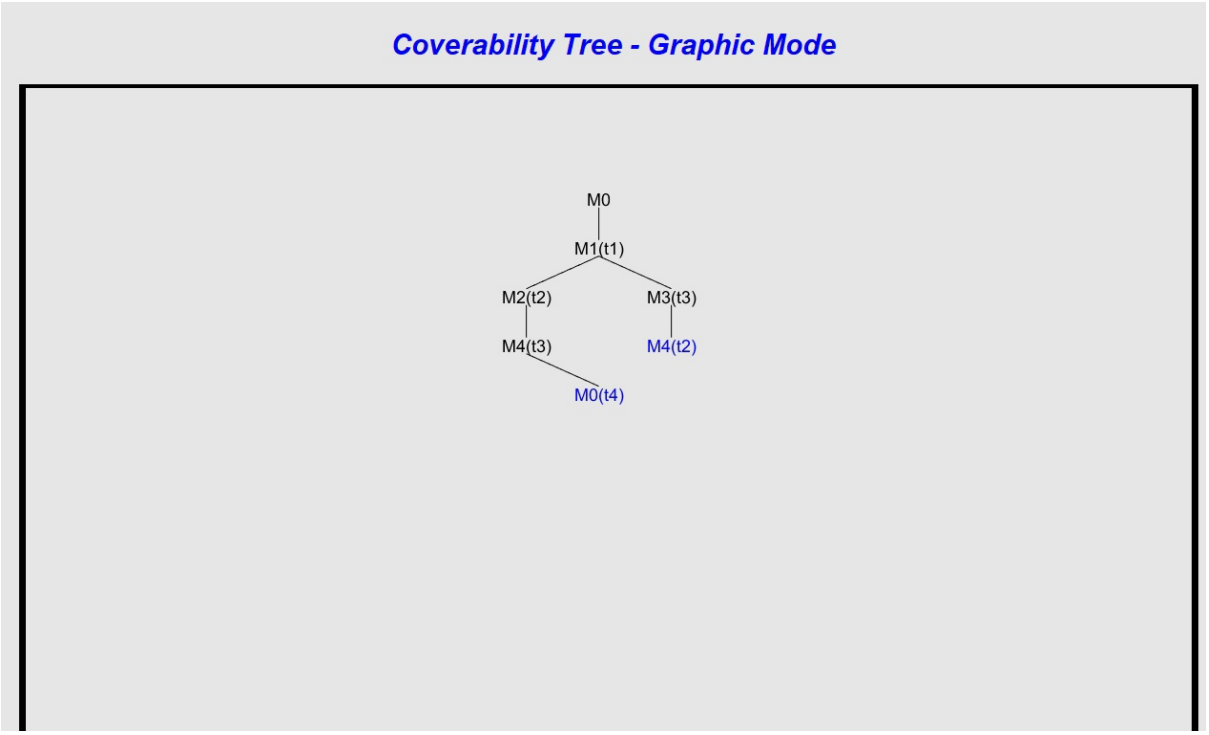


Диаграмма маркировок(Матлаб):

Coverability Tree - Graphic Mode



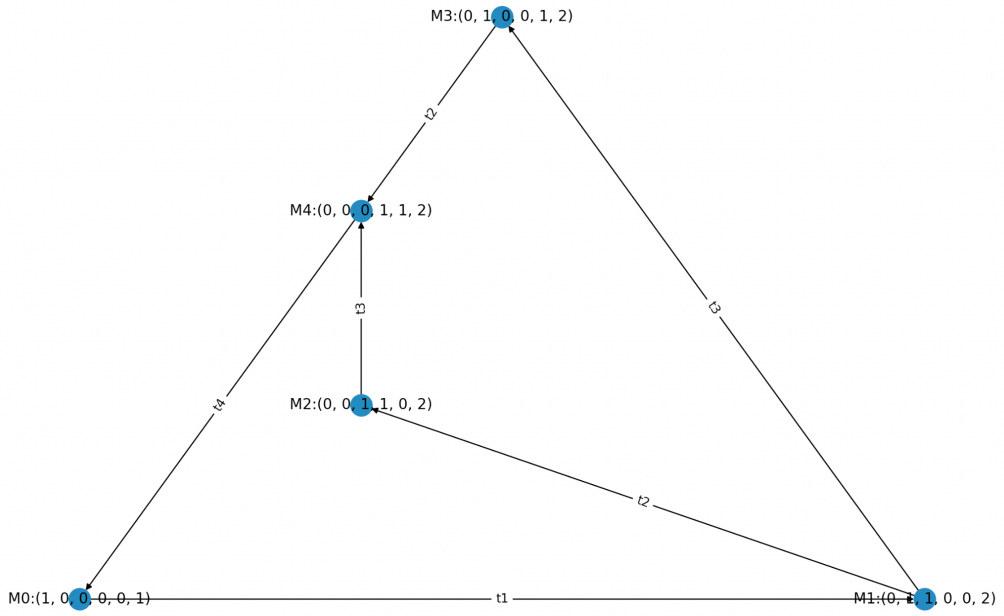
Coverability Tree - Text Mode

From	Fired	To	From	Fired	To	From	Fired	To	From	Fired	To	From	Fired	To	From	Fired	To
M0	t1	M1	**M1**	**t2**	**M2**	M1	t3	M3	**M2**	**t3**	**M4**	M3	t2	M4	**M4**	**t4**	**M0**

M[p1,p2,p3,p4,p5,p6]

M0 = [1,0,0,0,0,1]	M1 = [0,1,1,0,0,2]	M2 = [0,0,1,1,0,2]
M3 = [0,1,0,0,1,2]	M4 = [0,0,0,1,1,2]	

Диаграмма программы



Анализ сети

- Сеть является живой, т.к. каждый переход является потенциально срабатывающим при любой маркировке
- Сеть является ограниченной, k = 2
- Сеть не является безопасной, так как ограничена и k = 2.
- Сеть не является консервативной, т.к сумма фишек в состоянии не постоянна
- Сеть является маркированным графом,т.к.каждая позиция имеет в точности по одному входному и одному выходному переходу.

```
def is_marked_graph(D_input,D_output):
    for i in range(np.shape(D_input)[1]):
        if sum(D_input[:,i]) !=1 or sum(D_output[:,i])!=1:
            return False
    return True
```

Тип сети согласно матлаб