Row & Col Dominance

과 목 논리회로설계

전 공

학 번

이 름

목 차

Ⅰ. QM_Method의 구성요소

1. initial input
2. findPi & findEpi

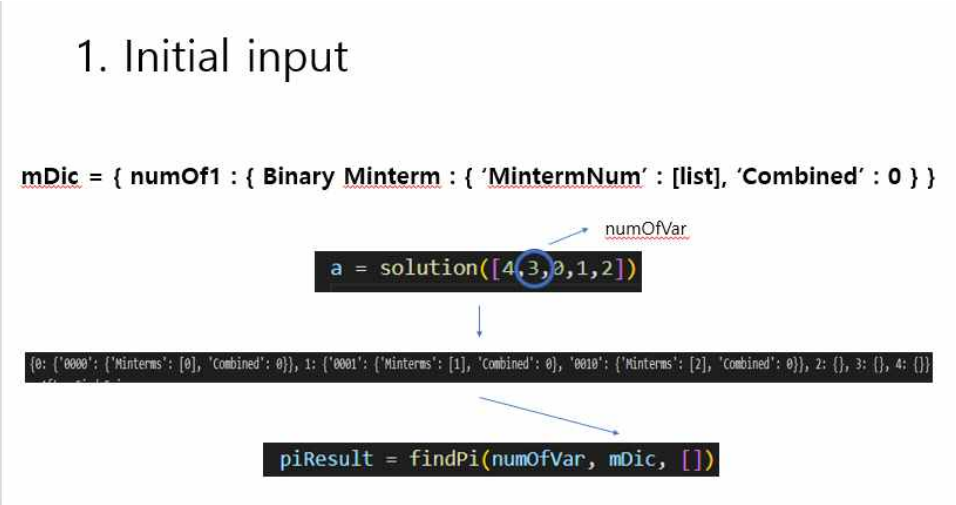
Ⅱ. Dominance Code

1. row domainance
2. column dominance

Ⅲ. Dominanace demonstration

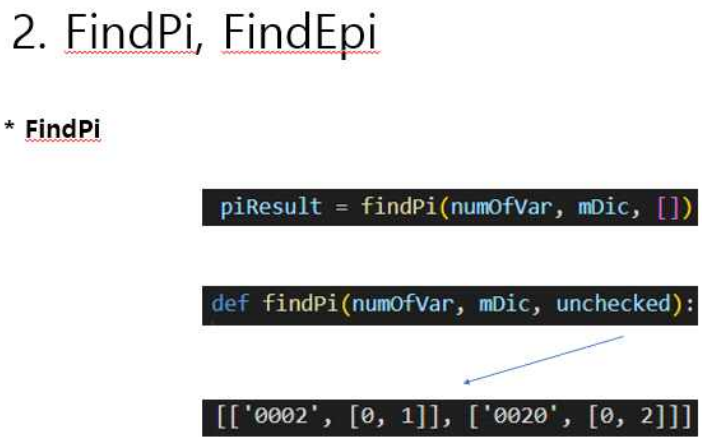
Ⅰ. QM_Method의 구성요소

1. initial input



초기 입력값은 mDic의 딕셔너리이다.
numOf1: minterm의 1의 개수
Binary Minterm: minterm의 binary 코드
'MintermNum': cover하는 minterm
'Combined': 해당 minterm이 combined되었는지 여부 판단

2. findPi & findEpi



매개변수: numOfVar(변수의 개수). mDic, unchecked(checked되지 않는 minterm 혹은 pi)

2. FindPi, FindEpi

*** FindEpi**

```
[['0002', [0, 1]], ['0020', [0, 2]]]
```

```
epiResult = findEpi(numOfVar,piResult)
```

```
--After Find Epi--
mCntDic: {}
epiList: [['0002', [0, 1]], ['0020', [0, 2]]]
nonEpiList: []
```

매개변수: numOfVar(변수의 개수). piResult(findPi 함수의 return값)

```
# dominance
while (True):
    dLen = len(mCntDic)
    if(dLen == 0):
        break

    changed = False # 두 dominance를 수행한 후에도 changed가 False라면, 더이상 epi를 찾을 없다는 뜻이다

    epiList += colDominance(nonEpiList, mCntDic)
    if (len(mCntDic) < dLen):
        dLen = len(mCntDic)
        changed = True
    print("--After Column Dominance--")
    # print("mCntDic: {0}\nepiList: {1}\nnonEpiList: {2}\n".format(mCntDic, epiFormat(numOfVar, epiList), epiFormat(numOfVar, nonEpiList)))
    print("mCntDic: {0}\nepiList: {1}\nnonEpiList: {2}\n".format(mCntDic, epiList, nonEpiList))

    epiList += rowDominance(nonEpiList, mCntDic)
    if (len(mCntDic) < dLen):
        dLen = len(mCntDic)
        changed = True

    print("--After Row Dominance--")
    # print("mCntDic: {0}\nepiList: {1}\nnonEpiList: {2}\n".format(mCntDic, epiFormat(numOfVar, epiList), epiFormat(numOfVar, nonEpiList)))
    print("mCntDic: {0}\nepiList: {1}\nnonEpiList: {2}\n".format(mCntDic, epiList, nonEpiList))

    if(not changed):
        break
```

findEpi 함수 내부에서 dominance 함수 실행

Ⅱ. Dominance Code

<dominance 함수의 매개변수>

```
(nonEpiList, mCntDic)
```

1. row dominance

<지역변수>

```
secondEpi = [] # 지배하는 pi
interchange = []
```

<동작 순서>

(1) nonEpiList에서 두 pi를 비교 대상으로 설정

(2) 두 번째 pi가 첫 번째 pi를 지배하는지 확인

```
# nonEpi들을 비교해서 dominance 관계인지 확인
for i in range(len(nonEpiList)-1):
    for j in range(i+1, len(nonEpiList)):

        # pi 1 < pi 2 인지 판별
        isDom = True
        for nonE in nonEpiList[i][1]:
            if (nonE not in nonEpiList[j][1]): # 첫번째 pi의 mintermNum이 두번째 pi에 없으면 break
                isDom = False
                break
```

(3) 만약 지배관계가 있으면 두 번째 pi를 secondEpi에 추가

(4) 만약 두 pi가 interchange한 관계인 경우에는 interchange list에 해당 pi가 cover하는 minterm들의 list를 저장해두고 마지막에 epi를 고른 후 nonEpiList에 남아있는 pi들을 제거해준다

```
if (isDom): # dominance 관계이면 list에 추가
    # 만약 이전 비교에서 첫 번째 pi가 지배했지만, 현재 비교에서는 지배당하는 경우
    # (더 큰 pi가 나타난 경우) secondEpi에서 첫 번째 pi를 삭제

    # interchange한 케이스일 때는 일단 interchange에 저장만 해둠
    if(nonEpiList[i][1] == nonEpiList[j][1]):
        interchange.append(nonEpiList[i][1])
        for minterm in nonEpiList[i][1]:
            mCntDic[minterm] = -1
        continue

    if(nonEpiList[i] in secondEpi):
        secondEpi.remove(nonEpiList[i])
    secondEpi.append(nonEpiList[j])
    for num in nonEpiList[j][1]:
        mCntDic[num] = -1
```

(5) 지배관계를 반대로 하여서 같은 방법을 사용하여 판별한다

```
# pi 1 > pi 2 인지 판별
for nonE in nonEpiList[j][1]:
    if (nonE not in nonEpiList[i][1]): # 두번째 pi의 mintermNum이 첫번째 pi에 없으면 break
        isDom = False
        break

if (isDom): # 지배관계가 반대로 형성되어 있을 때도 마찬가지
    if(nonEpiList[j] in secondEpi):
        secondEpi.remove(nonEpiList[j])
    secondEpi.append(nonEpiList[i])
    for num in nonEpiList[i][1]:
        mCntDic[num] = -1
```

(6) second epi를 nonEpiLIst에서 제거하고 interchange한 pi들을 해결한다

```
for epi in secondEpi:
    if(epi in nonEpiList):
        nonEpiList.remove(epi)

# findInterchange가 true로 변하는 시점의 pi만 epi로 쓰고 나머지는 버림
findInterchange = False
if interchange: #interchange한 pi들이 nonEpiList에 남아있으면
    for i in interchange:
        for nonEpi in nonEpiList:
            if(i == nonEpi[1]):
                if(findInterchange):
                    nonEpiList.remove(nonEpi)
                else:
                    secondEpi.append(nonEpi)
                    findInterchange = True
```

2. column dominance

<지역변수>

```
secondEpi = []
```

<동작과정>

(1) mCntDic에서 두 minterm을 비교 대상으로 설정한다

```
# minterm들을 비교해서 dominance 관계인지 확
for m1 in mCntDic:
    for m2 in mCntDic:
        if (m1 >= m2):
            continue
        if (mCntDic[m1] == -1):
            break
        if (mCntDic[m2] == -1):
            continue
```

(2) nonEpiList를 순회하며 pi가 m1 혹은 m2를 cover하고 있으면 각각의 set에 추가한다 (이때 각 set에는 해당 pi가 nonEpiList에 위치하고 있는 idx 값을 넣어준다)

```
pi_1 = set() # 첫 번째 minterm이 포함된 pi
pi_2 = set() # 두 번째 minterm이 포함된 pi

for nonEpi in nonEpiList:
    for m in nonEpi[1]:
        # nonEpi가 minterm 1 또는 2를 포함하고 있으면 각각의 set에 넣어줌
        if (m == m1):
            pi_1.add(nonEpiList.index(nonEpi))
        elif (m == m2):
            pi_2.add(nonEpiList.index(nonEpi))
```

(3) 첫 번째 pi가 두 번째 pi를 지배하는지 확인한다
(4) 만약 지배관계가 있으면 두 번째 minterm을 cover하는 pi들 중 for loop을 돌며 secondEpi에 없는 pi를 선택하여 secondEpi에 넣는다

```
# dominance 관계인지 확인 pi_1 > pi_2
isDom = True
for pi in pi_2:
    if (pi not in pi_1):
        isDom = False
        break

if (isDom):

    for idx in pi_2:
        # nonEpi 하나 고르기
        if (nonEpiList[idx] not in secondEpi): #secondEpi에 없는 pi를 선택
            # 그 nonEpi를 secondEpi에 넣기
            secondEpi.append(nonEpiList[idx])
```

(5) Second epi가 된 pi가 cover하고 있던 minterm들의 cnt를 -1로 변경하고 더불어 nonEpi에서도 같은 minterm을 cover하고 있다면 해당 minterm을 제거해준다(표에서 column을 줄이는 과정과 같다)

```
for m in nonEpiList[idx][1]:
    # epi가 포함하는 minterm들의 cnt를 -1로 변경
    mCntDic[m] = -1

    # nonEpi가 만약 없어질 minterm을 cover하고 있다면 제거해줌
    for nonEpi in nonEpiList:
        if (nonEpi == nonEpiList[idx]):
            continue
        if (m in nonEpi[1]):
            nonEpi[1].remove(m)
```

(6) nonEpiList에서 second epi가 된 pi를 제거해준다(표에서 row를 줄이는 과정과 같다)
(7) 만약 표를 줄이는 과정에서 추가적인 unchecked pi가 생긴다면 제거해준다

```
del nonEpiList[idx] # nonEpiList에서 pi 지우기

# 만약 모두 체크되지 않은 빈 pi가 추가적으로 생기면 제거
for nonEpi in nonEpiList:
    if(nonEpi[1] == []):
        nonEpiList.remove(nonEpi)
break
```

(8) 지배관계를 반대로 하여서 같은 방법을 사용한다

```
# dominance 관계인지 확인 pi_1 < pi_2
for pi in pi_1:
    if (pi not in pi_2):
        isDom = False
        break

if (isDom):

    for idx in pi_1:
        if (nonEpiList[idx] not in secondEpi):
            secondEpi.append(nonEpiList[idx])


            for m in nonEpiList[idx][1]:

                mCntDic[m] = -1

                for nonEpi in nonEpiList:
                    if (m in nonEpi[1]):
                        nonEpi[1].remove(m)

            del nonEpiList[idx]
            for nonEpi in nonEpiList:
                if(nonEpi[1] == []):
                    nonEpiList.remove(nonEpi)
            break
```

Ⅲ. Dominance demonstration

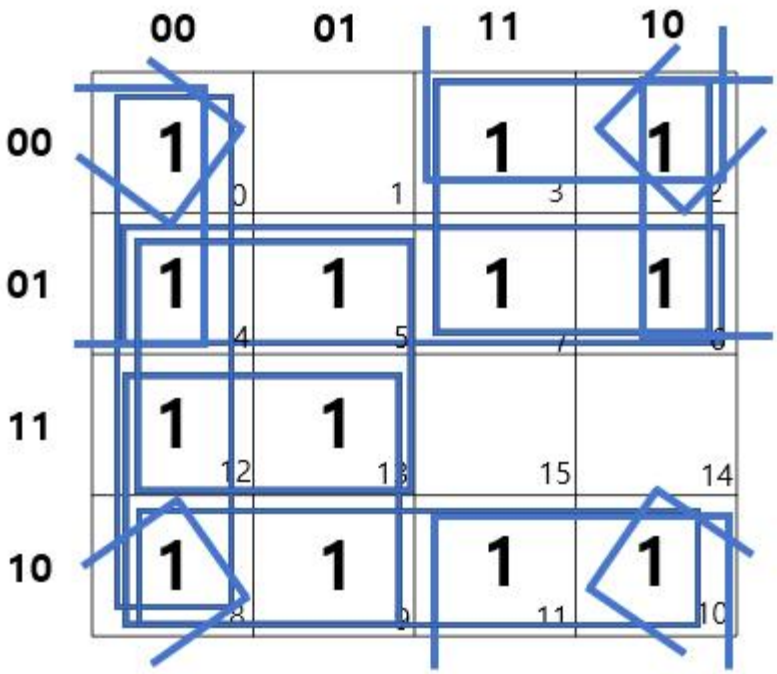
(1) solution에 다음의 값을 넣어주었다

```
b = solution([4, 13, 0, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13])
```

(2) findEpi까지 실행했을 때 epi가 발견되지 않았다

```
--After Find Epi--
mCntDic: {0: 3, 2: 4, 3: 2, 4: 4, 5: 2, 6: 3, 7: 2, 8: 4, 9: 2, 10: 3, 11: 2, 12: 3, 13: 2}
epiList: []
nonEpiList: [['0220', [0, 4, 2, 6]], ['2020', [0, 8, 2, 10]], ['2200', [0, 8, 4, 12]], ['0212', [2, 6, 3, 7]],
['2012', [2, 10, 3, 11]], ['0122', [4, 6, 5, 7]], ['2102', [4, 12, 5, 13]], ['1022', [8, 10, 9, 11]], ['1202',
[8, 12, 9, 13]]]
```

실제 k-map으로 따져 보았더니 epi가 발견되지 않았다



(3) nonEpi들에 대하여 column dominance를 적용해 보았다

column dominance 이전

	0	2	3	4	5	6	7	8	9	10	11	12	13
0--0	v	v		v		v							
-0-0	v	v						v		v			
--00	v			v				v				v	
0-1-		v	v			v	v						
-01-		v	v							v	v		
01--				v	v	v	v						
-10-				v	v							v	v
10--								v	v	v	v		
1-0-								v	v			v	v

이후

	0
0--0	v
-0-0	v
--00	v

출력값

```
--After Column Dominance--
mCntDic: {0: 3}
epiList: [['0212', [2, 6, 3, 7]], ['0122', [4, 5]], ['1022', [8, 10, 9, 11]], ['2102', [12, 13]]]
nonEpiList: [['0220', [0]], ['2020', [0]], ['2200', [0]]]
```

mCntDic에 0번 minterm만 남았고 3개의 nonEpi가 남은 것이 확인되었다

(4) nonEpi들에 대하여 row dominance를 적용해 보았다

```
nonEpiList: [['0220', [0]], ['2020', [0]], ['2200', [0]]]

--After Row Dominance--
mCntDic: {}
epiList: [['0212', [2, 6, 3, 7]], ['0122', [4, 5]], ['1022', [8, 10, 9, 11]], ['2102', [12, 13]]
, ['0220', [0]]]
nonEpiList: []
```

mCntDic의 길이가 0이 되었고, interchangalbe한 세 pi중에 하나가 선택되고 나머지는 삭제된 것을 확인할 수 있었다.

따라서 모든 코드가 정상 동작함을 확인하였다.