固定势集合的选择

李浩文、彼得·斯塔基

香港中文大學 The Chinese University of Hong Kong THE UNIVERSITY OF MELBOURNE

修改的集合选择问题 (baguaCard-10-8.dzn)

- 每个SYMB中的属性，给定一个数字1..nSpots的子集。选择一个大小为size的1..nSpots的子集，使得每个属性的子集中最多有一个元素在其中，并且最大化选择的集合的伤害值

```
nSpots = 10;
damage = [10, 8, 4, 2, 6, 9, 5, 3, 8, 10];
size = 3;
SYMB = {'天','泽','火','雷','风','水','山','地'};
group = [{1,4,6}, {1,2,6,7}, {1,3,6,8}, {1,2,3},
     {2,9,10}, {5,6,8,10}, {7,8,10}, {1,3,5}];
```

2

对修改的集合选择问题增加约束 (baguaCardSet.mzn)

- 增加的约束
  ```
  card(attacks) = size;
  ```
- 对模型求解
  ```
  attacks: {5,7,9} & damage: 19;
  ```
- 但是我们可以用不同方式对已知势的集合建模！

3

确定一个固定势集合

- 不去定义一个集合变量
  ```
  var set of SPOT: attacks;
  ```
  和使用势约束
  ```
  card(attacks) = size;
  ```
- 而是定义一个元素个数为size的数组
  ```
  array[1..size] of var SPOT: attacks;
  ```
  - 以及一些其他的约束 …
- 原因：假设 nSpots＝1000，size＝4
- 第一种表示方法：1000个布尔型变量
- 第二种表示方法：4个整型变量

4

确定一个固定势集合

- 观察
  `array[1..3] of var 1..10: x;`
- 代表多少个可能值？ 1000
- 以及 `var set of 1..10: x;` 和
  `card(x) = 3;`
  - 10 * 9 * 8 / 3 * 2 * 1 = 120
- 第一个问题：有一些数组表示并不是势为3的集合
  - 例如 [1,1,1] = {1}, [1,2,1] = {1,2}
- 解决方案：确保它们全部都不相同

```
forall(i,j in 1..u where i < j)
   (x[i] != x[j]);
```

5

确定一个固定势集合

- 观察

```
array[1..3] of var 1..10: x;
forall(i,j in 1..u where i < j
   (x[i] != x[j]);
```

- 代表多少个可能值？
  - 10 * 9 * 8 = 720
- 第二个问题：同一个集合有多重表示
  - 例如，{1,6,10} = [1,6,10], [10,1,6], [10,6,1], [1,10,6], [6,10,1], [6,1,10]
- 解决方案：确保按顺序排列

```
forall(i in 1..u-1)(x[i] < x[i+1]);
```

6

确定一个固定势集合

- 观察

```
array[1..3] of var 1..10: x;
forall(i,j in 1..u where i < j
    (x[i] != x[j]);
```

- 代表多少个可能值？
  - 10 * 9 * 8 = 720
- 第二个问题：同一个集合有多重表示
  - 例如，{1,6,10} = [1,6,10], [10,1,6], [10,6,1], [1,10,6], [6,10,1], [6,1,10]
- 解决方案：确保按顺序排列

```
forall(i in 1..u-1)(x[i] < x[i+1]);
```

7

决策变量建模中的关键问题

- 问题中的决策
  - 不一定是模型中的决策
- 如果可以就使之相同，但是
  - 假如你在决定
    - 一个图中的路径，
    - 一个树结构
- 建模的关键点（1）
  - （满足约束的）模型中的决策
  - 是问题中的有效决策
- 加约束来达到此目的
  - 例如，加约束强制数组元素 ≠

8

决策（变量）建模中的关键点

- **模型中的多个决策**
  - 代表问题中同一个的决策
  - 例如，x=[1,2,7]，x=[7,2,1] 代表了 x={1,2,7}
- **建模关键点（2）**
  - 对于每个解，模型中尽量只有一组决策与其对应
  - 代表了问题中的有效决策
- **加约束移除（冗余），只保留一组（决策表示）**
  - 例如，加约束强制数组元素 <

9

修改的集合选择问题 (baguaCard-10-8.dzn)

- **每个SYMB中的属性，给定一个数字1..nSpots的子集。选择一个大小为size的1..nSpots的子集，使得每个属性的子集中最多有一个元素在其中，并且最大化选择的集合的伤害值**

```
nSpots = 10;
damage = [10, 8, 4, 2, 6, 9, 5, 3, 8, 10];
size = 3;
SYMB = {'天','泽','火','雷','风','水','山','地'};
group = [{1,4,6}, {1,2,6,7}, {1,3,6,8}, {1,2,3},
     {2,9,10}, {5,6,8,10}, {7,8,10}, {1,3,5}];
```

10

八卦固定势模型 (baguaCardInt.mzn)

```
int: nSpots;
set of int: SPOT = 1..nSpots;
array[SPOT] of int: damage;
enum SYMB;
array[SYMB] of set of SPOT: group;
int: size;

array[1..size] of var SPOT: attacks;

constraint forall(i in 1..size-1)
   (attacks[i] < attacks[i+1]);
constraint forall(s in SYMB)(sum(i in 1..size)
   (attacks[i] in group[s]) <= 1);

var int: totalDamages =
   sum(i in 1..size)(damage[attacks[i]]);
solve maximize (totalDamages);
```

11

八卦固定势模型 (baguaCardInt.mzn)

```
int: nSpots;
set of int: SPOT = 1..nSpots;
array[SPOT] of int: damage;
enum SYMB;
array[SYMB] of set of SPOT: group;
int: size;

array[1..size] of var SPOT: attacks;

constraint forall(i in 1..size-1)
   (attacks[i] < attacks[i+1]);
constraint forall(s in SYMB)(sum(i in 1..size)
   (attacks[i] in group[s]) <= 1);

var int: totalDamages =
   sum(i in 1..size)(damage[attacks[i]]);
solve maximize (totalDamages);
```

决策变量

12

八卦固定势模型 (baguaCardInt.mzn)

```
int: nSpots;
set of int: SPOT = 1..nSpots;
array[SPOT] of int: damage;
enum SYMB;
array[SYMB] of set of SPOT: group;
int: size;

array[1..size] of var SPOT: attacks;

constraint forall(i in 1..size-1)
   (attacks[i] < attacks[i+1]);
constraint forall(s in SYMB)(sum(i in 1..size)
   (attacks[i] in group[s]) <= 1);

var int: totalDamages =
   sum(i in 1..size)(damage[attacks[i]]);
solve maximize (totalDamages);
```

有效表示

13

八卦固定势模型 (baguaCardInt.mzn)

```
int: nSpots;
set of int: SPOT = 1..nSpots;
array[SPOT] of int: damage;
enum SYMB;
array[SYMB] of set of SPOT: group;
int: size;

array[1..size] of var SPOT: attacks;

constraint forall(i in 1..size-1)
   (attacks[i] < attacks[i+1]);
constraint forall(s in SYMB)(sum(i in 1..size)
   (attacks[i] in group[s]) <= 1);

var int: totalDamages =
   sum(i in 1..size)(damage[attacks[i]]);
solve maximize (totalDamages);
```

交集最多只有一个元素

14

八卦固定势模型 (baguaCardInt.mzn)

```
int: nSpots;
set of int: SPOT = 1..nSpots;
array[SPOT] of int: damage;
enum SYMB;
array[SYMB] of set of SPOT: group;
int: size;

array[1..size] of var SPOT: attacks;

constraint forall(i in 1..size-1)
   (attacks[i] < attacks[i+1]);
constraint forall(s in SYMB)(sum(i in 1..size)
   (attacks[i] in group[s]) <= 1);

var int: totalDamages =
   sum(i in 1..size)(damage[attacks[i]]);
solve maximize (totalDamages);
```

目标

15

求解模型

- 对模型求解

```
attacks: [5,7,9] & damage: 19;
```

16

小结

- **有多种方式去表示固定**势集合
  - var set of OBJ + 势约束
    - 适用情况：求解器本身支持集合
    - 适用情况：OBJ不是太大
  - array[1..u] of var OBJ
    - 适用情况：当u比较小
- **两个**对决策变量建模时的关键问题（点）
  - **确保模型的每个解是**问题的一个解
  - **尽量确保**问题的每个解在模型中只有一个对应解（对称）

17

图像引用

所有图像由Marti Wong设计提供, © **香港中文大学与墨**尔本大**学** 2016

18