

# NOPT042 Constraint programming:

## Tutorial 9 - Implicit constraints

```
In [1]: %load_ext ipicat
```

Picat version 3.7

### Example: Seesaw

Adam, Boris, and Cecil want to sit on a 10-feet long seesaw such that they are at least 2 feet apart and the seesaw is balanced. Adam weighs 36 lbs, Boris 32 lbs, and Cecil 16 lbs. Write a general model. You can assume that the length is even, the distance is integer, and that they can only sit at integer points.

(Problem from Marriott & Stuckey "Programming with Constraints", page 257.  
Instance from R. Barták's tutorial.)

```
In [2]: !cat seesaw/instance1.pi
```

```
% sample instance
instance(NumPeople, Length, Distance, Weights) =>
    NumPeople = 3,
    Length = 10,
    Distance = 2,
    Weights = [36, 32, 16].
```

Possible decision variables?

- Position on the seesaw for each person.
- Distances between persons, position of the first person, and order of persons.
- Person or empty for each position on the seesaw.

Global constraints? Symmetry breaking? Multiple modeling? Search strategies?

```
In [3]: !ls seesaw/
```

```
instance1.pi  instance3.pi  instance5.pi  seesaw2.pi  seesaw4.pi
instance2.pi  instance4.pi  seesaw1.pi    seesaw3.pi
```

```
In [4]: !time picat seesaw/seesaw1.pi instance4.pi
!time picat seesaw/seesaw2.pi instance4.pi
!time picat seesaw/seesaw3.pi instance4.pi
!time picat seesaw/seesaw4.pi instance4.pi
```

```
[-16, -15, -14, -13, -12, -11, -10, -9, -8, -7, -6, -5, -4, -3, -2, -1, 0, 1, 2, 6, 8, 7, 9, 14, 15, 10, 16, 11, 12, 13]
```

```
real    0m26.653s
user    0m26.629s
sys      0m0.020s
```

```
[-16, -15, -14, -13, -12, -11, -10, -9, -8, -7, -6, -5, -4, -3, -2, -1, 0, 1, 2, 6, 8, 7, 9, 14, 15, 10, 16, 11, 12, 13]
```

```
real    0m28.832s
user    0m28.812s
sys      0m0.020s
```

```
[-16, -15, -14, -13, -12, -11, -10, -9, -8, -7, -6, -5, -4, -3, -2, -1, 0, 1, 2, 6, 8, 7, 9, 14, 15, 10, 16, 11, 12, 13]
```

```
real    0m41.612s
user    0m41.588s
sys      0m0.020s
```

```
[-8, 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, -7, -6, -5, -4, -3, -9, -10, -13, -12, -11, -16, -14]
```

```
real    0m0.033s
user    0m0.024s
sys      0m0.001s
```

## Example: Golomb's ruler

A [Golomb's ruler](#) is an imaginary ruler with  $n$  marks such that the distance between every two marks is different. Find the shortest possible ruler for a given  $n$ .

(The solution for  $N=28$  was announced on Nov 23, 2022! The length is 585.)

- What length are you able to solve in reasonable time?
- Add suitable implicit constraints. (We will discuss this in class.)

## Redundant (implicit) constraints

Redundant constraints do not restrict the solution set but rather express properties of a solution from a different viewpoint. This can lead to

- faster domain reduction,
- a significant boost in propagation,
- improved communication between variables.

We have already seen one example last week in the Magic sequence problem: adding the `scalar_product` constraint.

Implicit constraints based on the following:

$$dist[i, j] = dist[i, i + 1] + dist[i + 1, i + 2] + \dots + dist[j - 1, j]$$

Now estimate distances by 1, sum from i to j:

```
foreach(I in 1..N-1, J in I+1..N)
  Distances[I,J-I] #>= (J-I)*(J-I+1) div 2,
  Distances[I,J-I] #<= Length - (N-J+I-1)*(N-J+I) div 2
end
```

In [5]: !picat golomb/golomb.pi 10

CPU time 129.353 seconds. Backtracks: 14554575

```
length = 55
[0,1,6,10,23,26,34,41,53,55]
```

In [6]: !picat golomb/golomb-improved 11

CPU time 25.357 seconds. Backtracks: 1224484

```
length = 72
[0,1,4,13,28,33,47,54,64,70,72]
```

In [7]: !picat golomb/golomb-improved 11

CPU time 25.788 seconds. Backtracks: 1224484

```
length = 72
[0,1,4,13,28,33,47,54,64,70,72]
```