事例3（Car sequencing in Renault’s plants）

車両投入計画（塗装及びアセンブル）

- 従来はラインに沿って部品を配置する計画
- 同一色塗装の連続制約
- 二次的目的として色の変更を最小限にしたい

大規模な最適化問題

- **1300台の投入順**を決める → **400 000** バイナリ意志決定変数が必要
- MIP or CP ソルバーでは数時間の実行でも実行可能解さえ見つからない
- LocalSolver は**数秒**で実用的な精度を持つ解を Renault に提供

事例3

（LSP Model of Car sequencing in Renault）

$x_{cp} = 1 \Leftrightarrow$ **意志決定変数の定義（bool変数）**

```
x[1..nbClasses][1..nbPositions] <- bool();

for [c in 1..nbClasses]
   constraint sum[p in 1..nbPositions](x[c][p]) == card[c];

for [p in 1..nbPositions]
   constraint sum[c in 1..nbClasses](x[c][p]) == 1;

op[o in 1..nbOptions][p in 1..nbPositions] <- or[c in 1..nbClasses : options[c][o]](x[c][p]);

nbVehicles[o in 1..nbOptions][j in 1..nbPositions-Q[o]+1] <- sum[k in 1..Q[o]](op[o][j+k-1]);

violations[o in 1..nbOptions][j in 1..nbPositions-Q[o]+1] <- max(nbVehicles[o][j] - P[o], 0);

obj <- sum[o in 1..nbOptions][p in 1..nbPositions-Q[o]+1](violations[o][p]);
minimize obj;
```

事例3

（LSP Model of Car sequencing in Renault）

追加制約：同じ色の塗装は10台まで


```
color[p in 1..nbPositions] <- sum[c in 1..nbClasses](color[c] * x[c][p]);
same[p in 2..nbPositions] <- color[p] == color[p-1];
toolong[p in 11..nbPositions] <- and[j in p-10..p](same[j]);
for [p in 11..nbPositions] constraint not(toolong[p]);
```

意志決定変数は増やさず

追加目的関数：色切替回数を最小化

```
minimize sum[i in 2..nbPositions](not(same[i]));
```