On the in-the-middle algorithm and heuristic and some of its properties

Dag Wedelin, Data Science Division, CSE, Chalmers

Beginning with special cases of linear programming, I will describe these algorithms, and some of their properties. I will also briefly discuss the max-sum problem and related algorithms, and discuss some of the general challenges with numerical propagation algorithms in relation to classical constraint programming.
<table>
<thead>
<tr>
<th>Data</th>
<th>Plan</th>
<th>Rule</th>
<th>APC</th>
<th>Report</th>
<th>Options</th>
<th>Help</th>
</tr>
</thead>
<tbody>
<tr>
<td>01/11</td>
<td>02/11</td>
<td>03/11</td>
<td>04/11</td>
<td>05/11</td>
<td>06/11</td>
<td>07/11</td>
</tr>
<tr>
<td>12345</td>
<td>12346</td>
<td>12347</td>
<td>12348</td>
<td>04/04/11</td>
<td>04/05/11</td>
<td>04/06/11</td>
</tr>
<tr>
<td>MUC</td>
<td>FRA</td>
<td>HAM</td>
<td>DUS</td>
<td>HAM</td>
<td>DUS</td>
<td>HAM</td>
</tr>
<tr>
<td>FRA</td>
<td>HAM</td>
<td>DUS</td>
<td>HAM</td>
<td>DUS</td>
<td>HAM</td>
<td>DUS</td>
</tr>
<tr>
<td>HAM</td>
<td>DUS</td>
<td>HAM</td>
<td>DUS</td>
<td>HAM</td>
<td>DUS</td>
<td>HAM</td>
</tr>
<tr>
<td>DUS</td>
<td>HAM</td>
<td>DUS</td>
<td>HAM</td>
<td>DUS</td>
<td>HAM</td>
<td>DUS</td>
</tr>
<tr>
<td>HAM</td>
<td>DUS</td>
<td>HAM</td>
<td>DUS</td>
<td>HAM</td>
<td>DUS</td>
<td>HAM</td>
</tr>
<tr>
<td>DUS</td>
<td>HAM</td>
<td>DUS</td>
<td>HAM</td>
<td>DUS</td>
<td>HAM</td>
<td>DUS</td>
</tr>
<tr>
<td>HAM</td>
<td>DUS</td>
<td>HAM</td>
<td>DUS</td>
<td>HAM</td>
<td>DUS</td>
<td>HAM</td>
</tr>
<tr>
<td>DUS</td>
<td>HAM</td>
<td>DUS</td>
<td>HAM</td>
<td>DUS</td>
<td>HAM</td>
<td>DUS</td>
</tr>
<tr>
<td>HAM</td>
<td>DUS</td>
<td>HAM</td>
<td>DUS</td>
<td>HAM</td>
<td>DUS</td>
<td>HAM</td>
</tr>
<tr>
<td>DUS</td>
<td>HAM</td>
<td>DUS</td>
<td>HAM</td>
<td>DUS</td>
<td>HAM</td>
<td>DUS</td>
</tr>
<tr>
<td>HAM</td>
<td>DUS</td>
<td>HAM</td>
<td>DUS</td>
<td>HAM</td>
<td>DUS</td>
<td>HAM</td>
</tr>
<tr>
<td>DUS</td>
<td>HAM</td>
<td>DUS</td>
<td>HAM</td>
<td>DUS</td>
<td>HAM</td>
<td>DUS</td>
</tr>
<tr>
<td>HAM</td>
<td>DUS</td>
<td>HAM</td>
<td>DUS</td>
<td>HAM</td>
<td>DUS</td>
<td>HAM</td>
</tr>
<tr>
<td>DUS</td>
<td>HAM</td>
<td>DUS</td>
<td>HAM</td>
<td>DUS</td>
<td>HAM</td>
<td>DUS</td>
</tr>
<tr>
<td>HAM</td>
<td>DUS</td>
<td>HAM</td>
<td>DUS</td>
<td>HAM</td>
<td>DUS</td>
<td>HAM</td>
</tr>
<tr>
<td>DUS</td>
<td>HAM</td>
<td>DUS</td>
<td>HAM</td>
<td>DUS</td>
<td>HAM</td>
<td>DUS</td>
</tr>
<tr>
<td>HAM</td>
<td>DUS</td>
<td>HAM</td>
<td>DUS</td>
<td>HAM</td>
<td>DUS</td>
<td>HAM</td>
</tr>
<tr>
<td>DUS</td>
<td>HAM</td>
<td>DUS</td>
<td>HAM</td>
<td>DUS</td>
<td>HAM</td>
<td>DUS</td>
</tr>
<tr>
<td>HAM</td>
<td>DUS</td>
<td>HAM</td>
<td>DUS</td>
<td>HAM</td>
<td>DUS</td>
<td>HAM</td>
</tr>
<tr>
<td>DUS</td>
<td>HAM</td>
<td>DUS</td>
<td>HAM</td>
<td>DUS</td>
<td>HAM</td>
<td>DUS</td>
</tr>
<tr>
<td>HAM</td>
<td>DUS</td>
<td>HAM</td>
<td>DUS</td>
<td>HAM</td>
<td>DUS</td>
<td>HAM</td>
</tr>
<tr>
<td>DUS</td>
<td>HAM</td>
<td>DUS</td>
<td>HAM</td>
<td>DUS</td>
<td>HAM</td>
<td>DUS</td>
</tr>
<tr>
<td>HAM</td>
<td>DUS</td>
<td>HAM</td>
<td>DUS</td>
<td>HAM</td>
<td>DUS</td>
<td>HAM</td>
</tr>
<tr>
<td>DUS</td>
<td>HAM</td>
<td>DUS</td>
<td>HAM</td>
<td>DUS</td>
<td>HAM</td>
<td>DUS</td>
</tr>
<tr>
<td>HAM</td>
<td>DUS</td>
<td>HAM</td>
<td>DUS</td>
<td>HAM</td>
<td>DUS</td>
<td>HAM</td>
</tr>
<tr>
<td>DUS</td>
<td>HAM</td>
<td>DUS</td>
<td>HAM</td>
<td>DUS</td>
<td>HAM</td>
<td>DUS</td>
</tr>
<tr>
<td>HAM</td>
<td>DUS</td>
<td>HAM</td>
<td>DUS</td>
<td>HAM</td>
<td>DUS</td>
<td>HAM</td>
</tr>
<tr>
<td>DUS</td>
<td>HAM</td>
<td>DUS</td>
<td>HAM</td>
<td>DUS</td>
<td>HAM</td>
<td>DUS</td>
</tr>
<tr>
<td>HAM</td>
<td>DUS</td>
<td>HAM</td>
<td>DUS</td>
<td>HAM</td>
<td>DUS</td>
<td>HAM</td>
</tr>
<tr>
<td>DUS</td>
<td>HAM</td>
<td>DUS</td>
<td>HAM</td>
<td>DUS</td>
<td>HAM</td>
<td>DUS</td>
</tr>
<tr>
<td>HAM</td>
<td>DUS</td>
<td>HAM</td>
<td>DUS</td>
<td>HAM</td>
<td>DUS</td>
<td>HAM</td>
</tr>
<tr>
<td>DUS</td>
<td>HAM</td>
<td>DUS</td>
<td>HAM</td>
<td>DUS</td>
<td>HAM</td>
<td>DUS</td>
</tr>
</tbody>
</table>
In crew pairing system

- Generate many legal pairings
- Select an optimal subset of these pairings

Diagram:
- Generator
- Optimizer
- Solution
In crew pairing system

\[
\begin{align*}
\text{min } & cx \\
Ax & \geq 1 \quad \text{(set covering)} \\
Cx & \leq d \quad \text{(base capacity)} \\
x & \text{ binary vector}
\end{align*}
\]
paqs optimizer

- in-the-middle algorithm
- in-the-middle heuristic

In production since many years.

Regularly benchmarked, continuously improved.

Parallel implementation
the in-the-middle algorithm
The simple assignment problem

<table>
<thead>
<tr>
<th>persons</th>
<th>tasks</th>
</tr>
</thead>
<tbody>
<tr>
<td>3 8 6 4</td>
<td></td>
</tr>
<tr>
<td>4 2 3 0</td>
<td></td>
</tr>
<tr>
<td>2 + 10 8</td>
<td></td>
</tr>
<tr>
<td>6 8 10 5</td>
<td></td>
</tr>
</tbody>
</table>

\[
\begin{align*}
\text{max} & \quad c_1 x_1 + c_2 x_2 + \ldots + c_{16} x_{16} \\
\text{subject to} & \\
\begin{cases}
x_1 + x_2 + x_3 + x_4 = 1 \\
\vdots \\
x_1 + x_5 + x_9 + x_{13} = 1 \\
\vdots \\
x \text{ binary}
\end{cases}
\end{align*}
\]
\[
\begin{array}{ccc}
3 & 8 & 6 \\
4 & 2 & 3 \\
2 & 7 & 108 \\
6 & 8 & 105
\end{array}
\Rightarrow
\begin{array}{ccc}
-4 & 1 & -1 \\
4 & 2 & 3 \\
2 & 7 & 108 \\
6 & 8 & 105
\end{array}
\]
Subtract average of largest and second largest numbers.

Iterate for all rows and columns until there are no more sign changes.

Simplest possible algorithm? (just subtracting the smallest number does not work)

select assignments with positive cost!
In-the-middle for 0-1 ILP

\[
\begin{align*}
\max & \ c^T x \\
A x &= b \\
x &\text{ binary}
\end{align*}
\]

\[
A \text{ contains } \{-1,0,1\}
\]

\[
b \text{ integer inequalities ok}
\]

Consider

\[
\overline{c} = c - y A
\]

iteratively make single constraints feasible by selecting the dual \( y \) in-the-middle of the possible interval
A “dual” algorithm

Minimize piecewise linear convex function with coordinate descent

\[
\min_y f(y) = yb + \max_{0 \leq x \leq 1} \bar{c}x
\]
Theorem:

The in-the-middle algorithm can only solve “easy” ILP problems (the LP relaxation has a unique solution that is integer).

What about convergence?
How does in-the-middle fail for difficult ILP’s?

a lot of zeros…

eg. 8-queens with diagonal constraints
A small difficult problem

\[
\begin{align*}
\text{max} & \quad x_1 + x_2 + x_3 \\
\quad x_1 &= x_2 + x_3 \\
\quad x_2 &= x_1 + x_3 \\
\quad x_3 &= x_1 + x_3 \\
\overline{c}_1 &= \overline{c}_2 = \overline{c}_3 = 0 \text{ is a fixpoint!}
\end{align*}
\]
in-the-middle heuristic
for difficult 0-1 ILP's

invariant

not invariant!

\[ \text{proportional to } (r^+ - r^-) \]
Now it suddenly works!

(weighted 8-queens problem)
in-the-middle heuristic

\[ y_i = -\frac{r^+ + r^-}{2} \pm \alpha \frac{r^+-r^-}{2} \]

in-the-middle algorithm

in-the-middle heuristic
Sweep mechanism

best results with as low disturbance as possible

(also small random costs for resolving ties)
the max-sum problem
The max-sum problem

\[
\max_w f(w) = \sum_k g_k(w^k) + C
\]

- vector of discrete variables
- different subsets of the variables

Allows a highly non-linear cost function
Equivalent problems

\[
g(w_1) \quad g(w_1, w_2) \quad g(w_2) \quad g(w_2, w_3) \quad g(w_3)
\]

constraint components

variable components
modelling and solving the max-sum problem as an ILP
The **max-sum ILP**

For C, we introduce an extra variable \( x_C \), together with the constraint \( x_C = 1 \).
Max-sum with the in-the-middle algorithm

start

0
3
3
3

5
7
-5
7

9
-1
1
5
-7
5

in-the-middle

(makes linear constraint feasible)
Or solve entire subproblems with specialised algorithm!

Iteratively update one subproblem at a time
Move in and move out…

We want to keep the best combination still largest in constraint!

“non-conflicting” minimizes $f(y)$

move in variable components

move out, but not too much
What if we move out “too much”?

We can easily explain why the **max-sum algorithm** does not guarantee optimality!

Here we have moved out even more, we get a “conflict”!

(max-sum algorithm!)
Many interesting relationships between algorithms!

- Convergent:
  - In-the-middle algorithm
  - Generalised iterative scaling
  - Max-sum diffusion

- Non-convergent:
  - Max-sum heuristic
  - Loopy belief propagation
  - Tree-reweighted message passing
Can we unify the models?

\[
\max_w f(w) = \sum_k g_k(w^k) + C
\]

\[
\begin{align*}
\max c x \\
Ax &= b \\
x_j &\in \{0, 1\}
\end{align*}
\]
Let's model the other way around: ILP to max-sum!

\[
\max \{2x_1 + 3x_2 + 2x_3 \mid x_1 + x_2 = 1, x_2 + x_3 = 1, x_j \text{ binary}\}.
\]

\[
\max \ g_1(x_1) + g_2(x_2) + g_3(x_3) + g_4(x_1, x_2) + g_5(x_2, x_3).
\]
\[
\max_w f(w) = \sum_k g_k(w^k) + C
\]

\[
\begin{align*}
\max c x \\
Ax &= b \\
x_j &\in \{0, 1\}
\end{align*}
\]

The in-the-middle updates can be seen as fast specialized max-sum constraint updates!

The distinction between an ILP model and a max-sum model is blurred!

The in-the-middle algorithm for the original ILP

\[
\text{same as in-the-middle heuristic (for } \alpha = 1)\]

if we move out as much as possible but not too much…

\[
\Rightarrow
\]

same as in-the-middle algorithm for the original ILP
To note from a constraint programming perspective

**Numerical propagation** can solve many non-trivial problems with propagation only!

(no combinatorial search!)
Summary

in-the-middle algorithm
in-the-middle heuristic

unify algorithms!

unify models!
END