

Davi Doro
davi_doro@hotmail.com

Ricardo Camargo
rcamargo@dep.ufmg.br

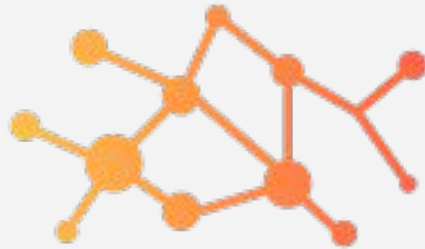


LaModAI



Davi Doro
davi_doro@hotmail.com

Ricardo Camargo
rcamargo@dep.ufmg.br

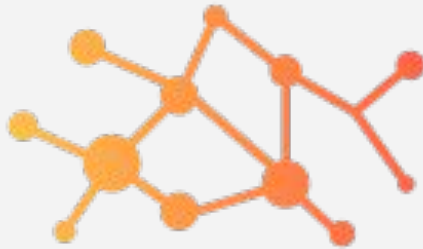


LaModAI

Davi Doro
davi_doro@hotmail.com



Ricardo Camargo
rcamargo@dep.ufmg.br

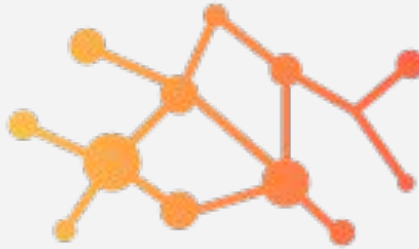


LaModAI



Davi Doro
davi_doro@hotmail.com

Ricardo Camargo
rcamargo@dep.ufmg.br



LaModAI

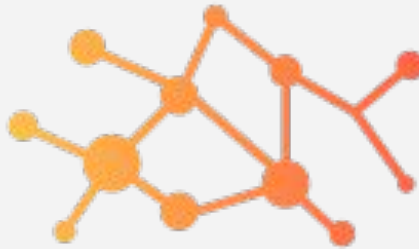
Laboratório de **M**odelagem **A**lgébrica

Davi Doro

davi_doro@hotmail.com

Ricardo Camargo

rcamargo@dep.ufmg.br



LaModAI

Algebraic Modeling Laboratory

Davi Doro

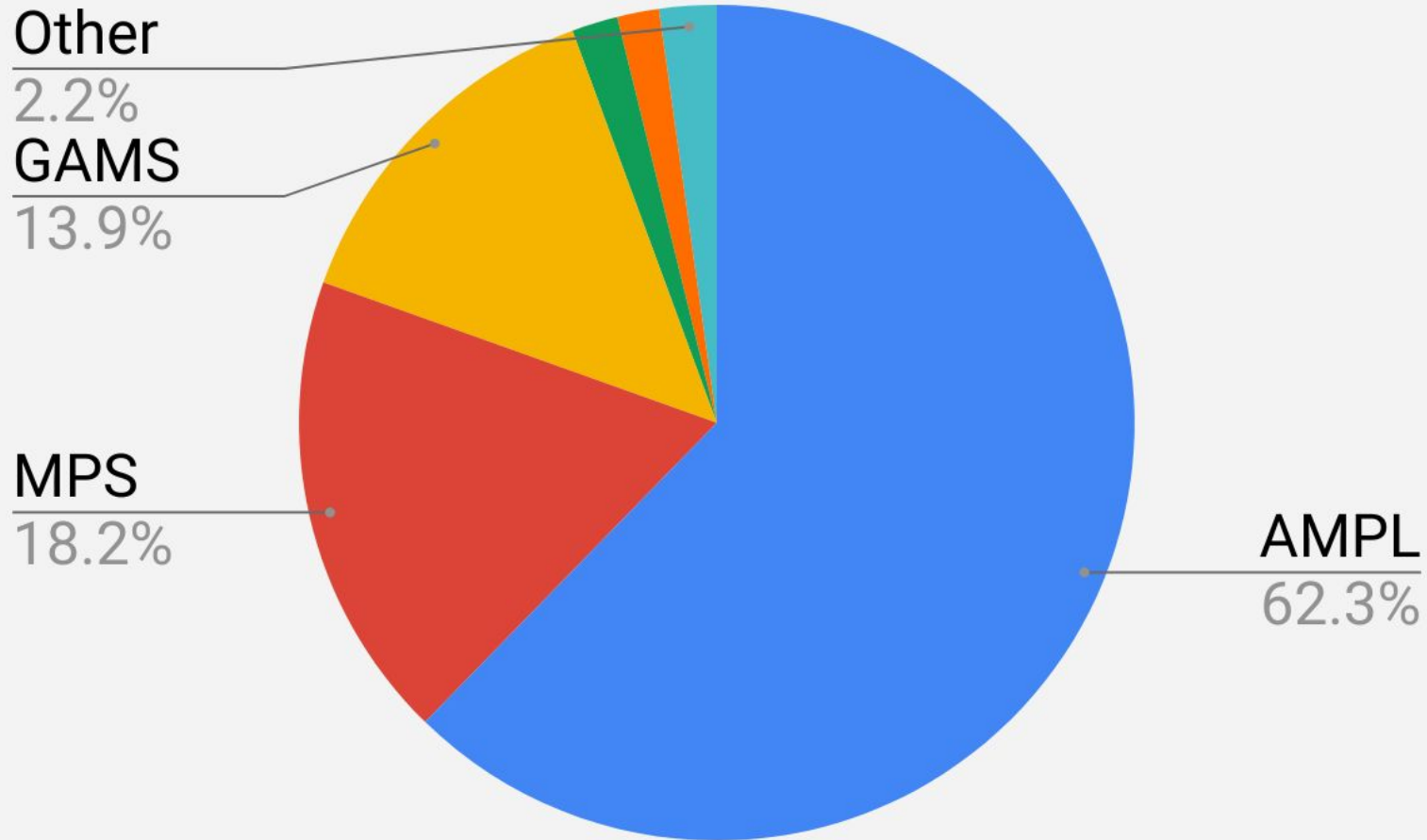
davi_doro@hotmail.com

Ricardo Camargo

rcamargo@dep.ufmg.br



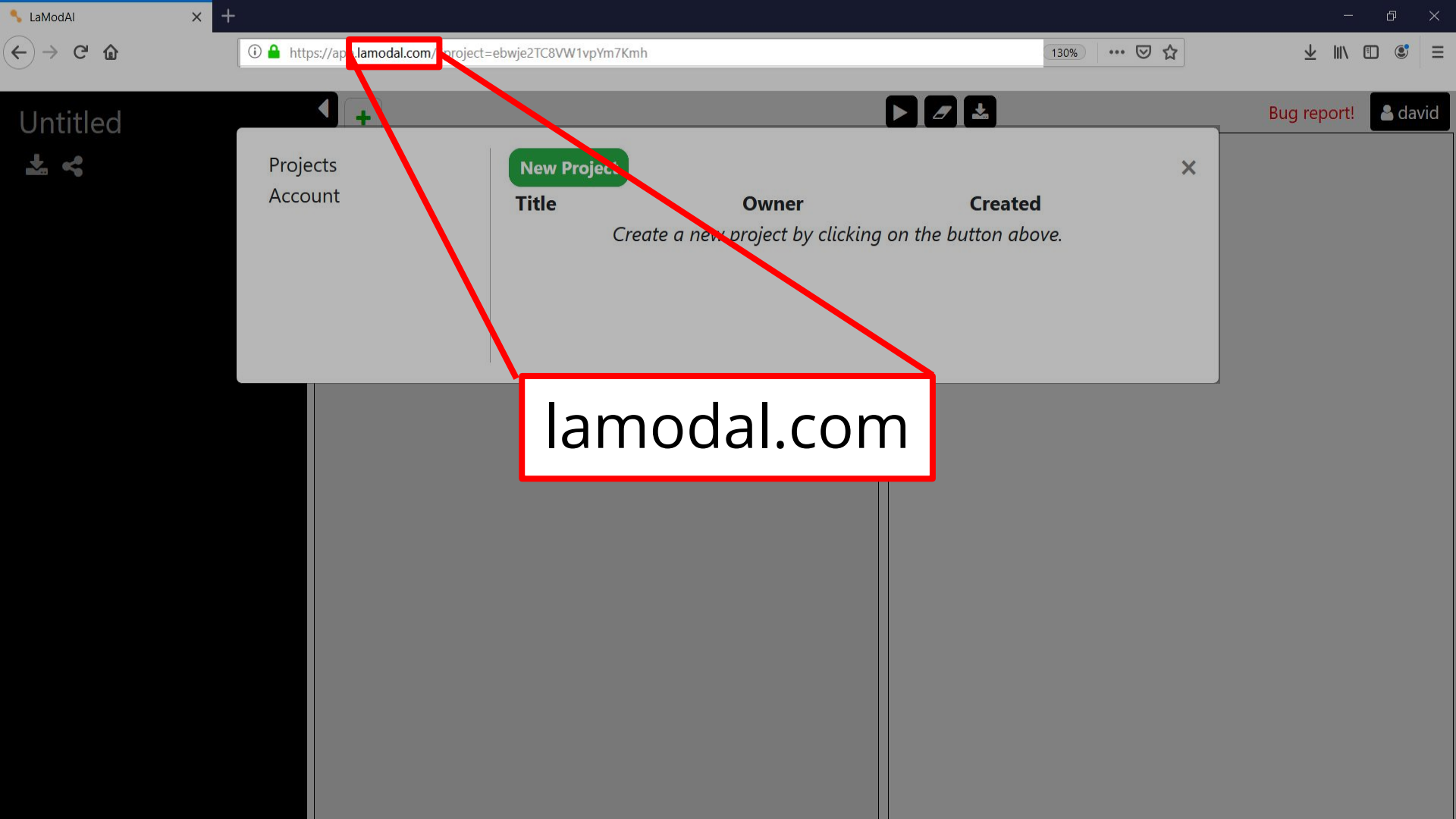
NEOS: Preferred languages



Projects
Account

New Project X

Title	Owner	Created
<i>Create a new project by clicking on the button above.</i>		



lamodal.com

lamodal.com

Project title: ✕

Job Shop

Optional - drop files below:

File Name	Size	
wagner.mod	1496b	✕
manne.mod	1054b	✕
pcp.run	30b	✕
small-instance.dat	591b	✕

Create Cancel

Project title: ✕

Optional - drop files below:

File Name	Size	
wagner.mod	1496b	✕
manne.mod	1054b	✕
pcp.run	30b	✕
small-instance.dat	591b	✕



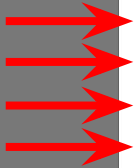
Project title: ×

Job Shop

Optional - drop files below:

File Name	Size	
wagner.mod	1496b	×
manne.mod	1054b	×
pcp.run	30b	×
small-instance.dat	591b	×

Create Cancel



Job Shop



wagner.mod

pcp.run

manne.mod

small-instance.dat

wagner.mod



```

1 param n_jobs, integer, > 0;
2 param n_operations, integer, > 0;
3 param n_machines, integer, > 0;
4
5 set N;
6 set M;
7 set J := 1..n_jobs;
8 set job_operations{J};
9 set operations_at{M};
10 set A, dimen 2;
11 set E, dimen 2;
12
13 param p{N}, >= 0;
14 param m{N}, integer, > 0;
15 param n{M}, integer, > 0;
16 param inf;
17
18 var y{i in N, 1..n[m[i]]}, binary;
19 var t{k in M, 1..n[k]}, >= 0;
20 var s{k in M, 1..n[k]}, >= 0;
21 var s0{k in M}, >= 0;
22 var T{k in M, 1..n[k]}, >= 0;
23 var c_max, >= 0;
24
25 minimize objective: c_max;
26
27 s.t. r1{k in M, l in 1..n[k]}: sum{i in operations
28 s.t. r2{i in N}: sum{l in 1..n[m[i]]} y[i, l] = 1;
29 s.t. r3{k in M, l in 1..n[k]}: T[k, l] = sum{i in
30 s.t. r4{(i, j) in A, li in 1..n[m[i]], lj in 1..n[
31 s.t. r5{k in M}: t[k, 1] = s0[k];
32 s.t. r6{k in M, r in 2..n[k]}: t[k, r] = s0[k] + s
33 s.t. r7{k in M}: c_max >= t[k, n[k]] + T[k, n[k]];

```



Bug report!

david

```

> help
NEOS-server usage:

neos exec|submit <model> <data> <script>

Examples:
neos exec my-model.mod my-data.dat my-script.run
neos submit my-model.mod my-data.dat my-script.run

Note:
- Use 'exec' for jobs that take no longer than 5 minutes.
  Results will be streamed here.
- Use 'submit' for jobs longer than 5 minutes. You will
  receive the results by email.

-----
Other commands:

clear : to clear the screen.
help  : to see this message.
>

```

Job Shop

- wagner.mod
- pcp.run
- manne.mod
- small-instance.dat

```
wagner.mod +
1 param n_jobs, integer, > 0;
2 param n_operations, integer, > 0;
3 param n_machines, integer, > 0;
4
5 set N;
6 set M;
7 set J := 1..n_jobs;
8 set job_operations{J};
9 set operations_at{M};
10 set A, dimen 2;
11 set E, dimen 2;
12
13 param p{N}, >= 0;
14 param m{N}, integer, > 0;
15 param n{M}, integer, > 0;
16 param inf;
17
18 var y{i in N, 1..n[m[i]]}, binary;
19 var t{k in M, 1..n[k]}, >= 0;
20 var s{k in M, 1..n[k]}, >= 0;
21 var s0{k in M}, >= 0;
22 var T{k in M, 1..n[k]}, >= 0;
23 var c_max, >= 0;
24
25 minimize objective: c_max;
26
27 s.t. r1{k in M, l in 1..n[k]}: sum{i in operations
28 s.t. r2{i in N}: sum{l in 1..n[m[i]]} y[i, l] = 1;
29 s.t. r3{k in M, l in 1..n[k]}: T[k, l] = sum{i in
30 s.t. r4{(i, j) in A, li in 1..n[m[i]], lj in 1..n[
31 s.t. r5{k in M}: t[k, 1] = s0[k];
32 s.t. r6{k in M, r in 2..n[k]}: t[k, r] = s0[k] + s
33 s.t. r7{k in M}: c_max >= t[k, n[k]] + T[k, n[k]];
```

```
> help
NEOS-server usage:

neos exec|submit <model> <data> <script>

Examples:
neos exec my-model.mod my-data.dat my-script.run
neos submit my-model.mod my-data.dat my-script.run

Note:
- Use 'exec' for jobs that take no longer than 5 minutes.
  Results will be streamed here.
- Use 'submit' for jobs longer than 5 minutes. You will
  receive the results by email.

-----
Other commands:

clear : to clear the screen.
help  : to see this message.
>
```

- wagner.mod
- pcp.run
- manne.mod
- small-instance.dat

```
wagner.mod +
1 param n_jobs, integer, > 0;
2 param n_operations, integer, > 0;
3 param n_machines, integer, > 0;
4
5 set N;
6 set M;
7 set J := 1..n_jobs;
8 set job_operations{J};
9 set operations_at{M};
10 set A, dimen 2;
11 set E, dimen 2;
12
13 param p{N}, >= 0;
14 param m{N}, integer, > 0;
15 param n{M}, integer, > 0;
16 param inf;
17
18 var y{i in N, 1..n[m[i]]}, binary;
19 var t{k in M, 1..n[k]}, >= 0;
20 var s{k in M, 1..n[k]}, >= 0;
21 var s0{k in M}, >= 0;
22 var T{k in M, 1..n[k]}, >= 0;
23 var c_max, >= 0;
24
25 minimize objective: c_max;
26
27 s.t. r1{k in M, l in 1..n[k]}: sum{i in operations
28 s.t. r2{i in N}: sum{l in 1..n[m[i]]} y[i, l] = 1;
29 s.t. r3{k in M, l in 1..n[k]}: T[k, l] = sum{i in
30 s.t. r4{(i, j) in A, li in 1..n[m[i]], lj in 1..n[
31 s.t. r5{k in M}: t[k, 1] = s0[k];
32 s.t. r6{k in M, r in 2..n[k]}: t[k, r] = s0[k] + s
33 s.t. r7{k in M}: c_max >= t[k, n[k]] + T[k, n[k]];
```

```
> help
NEOS-server usage:

neos exec|submit <model> <data> <script>

Examples:
neos exec my-model.mod my-data.dat my-script.run
neos submit my-model.mod my-data.dat my-script.run

Note:
- Use 'exec' for jobs that take no longer than 5 minutes.
  Results will be streamed here.
- Use 'submit' for jobs longer than 5 minutes. You will
  receive the results by email.

-----
Other commands:

clear : to clear the screen.
help  : to see this message.
>
```


Job Shop

- wagner.mod
- pcp.run
- manne.mod
- small-instance.dat

wagner.mod

```

1 param n_jobs, integer, > 0;
2 param n_operations, integer, > 0;
3 param n_machines, integer, > 0;
4
5 set N;
6 set M;
7 set J := 1..n_jobs;
8 set job_operations{J};
9 set operations_at{M};
10 set A, dimen 2;
11 set E, dimen 2;
12
13 param p{N}, >= 0;
14 param m{N}, integer, > 0;
15 param n{M}, integer, > 0;
16 param inf;
17
18 var y{i in N, 1..n[m[i]]}, binary;
19 var t{k in M, 1..n[k]}, >= 0;
20 var s{k in M, 1..n[k]}, >= 0;
21 var s0{k in M}, >= 0;
22 var T{k in M, 1..n[k]}, >= 0;
23 var c_max, >= 0;
24
25 minimize objective: c_max;
26
27 s.t. r1{k in M, l in 1..n[k]}: sum{i in operations
28 s.t. r2{i in N}: sum{l in 1..n[m[i]]} y[i, l] = 1;
29 s.t. r3{k in M, l in 1..n[k]}: T[k, l] = sum{i in
30 s.t. r4{(i, j) in A, li in 1..n[m[i]], lj in 1..n[
31 s.t. r5{k in M}: t[k, 1] = s0[k];
32 s.t. r6{k in M, r in 2..n[k]}: t[k, r] = s0[k] + s
33 s.t. r7{k in M}: c_max >= t[k, n[k]] + T[k, n[k]];

```

Bug report! david

```

> help
NEOS-server usage:

neos exec|submit <model> <data> <script>

Examples:
neos exec my-model.mod my-data.dat my-script.run
neos submit my-model.mod my-data.dat my-script.run

Note:
- Use 'exec' for jobs that take no longer than 5 minutes.
  Results will be streamed here.
- Use 'submit' for jobs longer than 5 minutes. You will
  receive the results by email.

-----
Other commands:

clear : to clear the screen.
help  : to see this message.
>

```

Job Shop

- wagner.mod
- pcp.run
- manne.mod
- small-instance.dat

wagner.mod +

```
1 param n_jobs, integer, > 0;
2 param n_operations, integer, > 0;
3 param n_machines, integer, > 0;
4
5 set N;
6 set M;
7 set J := 1..n_jobs;
8 set job_operations{J};
9 set operations_at{M};
10 set A, dimen 2;
11 set E, dimen 2;
12
13 param p{N}, >= 0;
14 param m{N}, integer, > 0;
15 param n{M}, integer, > 0;
16 param inf;
17
18 var y{i in N, 1..n[m[i]]}, binary;
19 var t{k in M, 1..n[k]}, >= 0;
20 var s{k in M, 1..n[k]}, >= 0;
21 var s0{k in M}, >= 0;
22 var T{k in M, 1..n[k]}, >= 0;
23 var c_max, >= 0;
24
25 minimize objective: c_max;
26
27 s.t. r1{k in M, l in 1..n[k]}: sum{i in operations
28 s.t. r2{i in N}: sum{l in 1..n[m[i]]} y[i, l] = 1;
29 s.t. r3{k in M, l in 1..n[k]}: T[k, l] = sum{i in
30 s.t. r4{(i, j) in A, li in 1..n[m[i]], lj in 1..n[
31 s.t. r5{k in M}: t[k, 1] = s0[k];
32 s.t. r6{k in M, r in 2..n[k]}: t[k, r] = s0[k] + s
33 s.t. r7{k in M}: c_max >= t[k, n[k]] + T[k, n[k]];
```

 Bug report! david

```
> help
NEOS-server usage:

neos exec|submit <model> <data> <script>

Examples:
neos exec my-model.mod my-data.dat my-script.run
neos submit my-model.mod my-data.dat my-script.run

Note:
- Use 'exec' for jobs that take no longer than 5 min
  Results will be streamed here.
- Use 'submit' for jobs longer than 5 minutes. You w
  receive the results by email.

-----
Other commands:

clear : to clear the screen.
help : to see this message.
> neos exec wagner.mod small-instance.dat pcp.run|
```

- wagner.mod
- pcp.run
- manne.mod
- small-instance.dat

```
1 param n_jobs, integer, > 0;
2 param n_operations, integer, > 0;
3 param n_machines, integer, > 0;
4
5 set N;
6 set M;
7 set J := 1..n_jobs;
8 set job_operations{J};
9 set operations_at{M};
10 set A, dimen 2;
11 set E, dimen 2;
12
13 param p{N}, >= 0;
14 param m{N}, integer, > 0;
15 param n{M}, integer, > 0;
16 param inf;
17
18 var y{i in N, 1..n[m[i]]}, binary;
19 var t{k in M, 1..n[k]}, >= 0;
20 var s{k in M, 1..n[k]}, >= 0;
21 var s0{k in M}, >= 0;
22 var T{k in M, 1..n[k]}, >= 0;
23 var c_max, >= 0;
24
25 minimize objective: c_max;
26
27 s.t. r1{k in M, l in 1..n[k]}: sum{i in operations
28 s.t. r2{i in N}: sum{l in 1..n[m[i]]} y[i, l] = 1;
29 s.t. r3{k in M, l in 1..n[k]}: T[k, l] = sum{i in
30 s.t. r4{(i, j) in A, li in 1..n[m[i]], lj in 1..n[
31 s.t. r5{k in M}: t[k, 1] = s0[k];
32 s.t. r6{k in M, r in 2..n[k]}: t[k, r] = s0[k] + s
33 s.t. r7{k in M}: c_max >= t[k, n[k]] + T[k, n[k]];
```

```
> help
NEOS-server usage:

neos exec|submit <model> <data> <script>

Examples:
neos exec my-model.mod my-data.dat my-script.run
neos submit my-model.mod my-data.dat my-script.run

Note:
- Use 'exec' for jobs that take no longer than 5 minutes.
  Results will be streamed here.
- Use 'submit' for jobs longer than 5 minutes. You will
  receive the results by email.

-----
Other commands:

clear : to clear the screen.
help : to see this message.
> neos exec wagner.mod small-instance.dat pcp.run
Submission successful
Job number : 7214152
Job password: tiIqMKcN
waiting for job dispatch...
|
```

Job Shop

- wagner.mod
- pcp.run
- manne.mod
- small-instance.dat

wagner.mod +

```

1 param n_jobs, integer, > 0;
2 param n_operations, integer, > 0;
3 param n_machines, integer, > 0;
4
5 set N;
6 set M;
7 set J := 1..n_jobs;
8 set job_operations{J};
9 set operations_at{M};
10 set A, dimen 2;
11 set E, dimen 2;
12
13 param p{N}, >= 0;
14 param m{N}, integer, > 0;
15 param n{M}, integer, > 0;
16 param inf;
17
18 var y{i in N, 1..n[m[i]]}, binary;
19 var t{k in M, 1..n[k]}, >= 0;
20 var s{k in M, 1..n[k]}, >= 0;
21 var s0{k in M}, >= 0;
22 var T{k in M, 1..n[k]}, >= 0;
23 var c_max, >= 0;
24
25 minimize objective: c_max;
26
27 s.t. r1{k in M, l in 1..n[k]}: sum{i in operations
28 s.t. r2{i in N}: sum{l in 1..n[m[i]]} y[i, l] = 1;
29 s.t. r3{k in M, l in 1..n[k]}: T[k, l] = sum{i in
30 s.t. r4{(i, j) in A, li in 1..n[m[i]], lj in 1..n[
31 s.t. r5{k in M}: t[k, 1] = s0[k];
32 s.t. r6{k in M, r in 2..n[k]}: t[k, r] = s0[k] + s
33 s.t. r7{k in M}: c_max >= t[k, n[k]] + T[k, n[k]];

```

▶ ✎ ⬇ Bug report! david

```







processing data.
processing commands.
Executing on prod-exec-1.neos-server.org

Presolve eliminates 3 constraints and 6 variables.
Adjusted problem:
33 variables:
    12 binary variables
    21 linear variables
45 constraints, all linear; 133 nonzeros
    25 equality constraints
    20 inequality constraints
1 linear objective; 1 nonzero.

CPLEX 12.7.0.0: threads=4
CPLEX 12.7.0.0: optimal integer solution; objective 2
22 MIP simplex iterations
0 branch-and-bound nodes
No basis.
objective = 20

t :=
1 1   0
1 2   5
2 1   0
2 2   5
3 1  10
4 1  10
4 2  15
;
> |

```

-  
-  wagner.mod
-  pcp.run
-  manne.mod
-  small-instance.dat

wagner.mod +

```

1 param n_jobs, integer, > 0;
2 param n_operations, integer, > 0;
3 param n_machines, integer, > 0;
4
5 set N;
6 set M;
7 set J := 1..n_jobs;
8 set job_operations{J};
9 set operations_at{M};
10 set A, dimen 2;
11 set E, dimen 2;
12
13 param p{N}, >= 0;
14 param m{N}, integer, > 0;
15 param n{M}, integer, > 0;
16 param inf;
17
18 var y{i in N, 1..n[m[i]]}, binary;
19 var t{k in M, 1..n[k]}, >= 0;
20 var s{k in M, 1..n[k]}, >= 0;
21 var s0{k in M}, >= 0;
22 var T{k in M, 1..n[k]}, >= 0;
23 var c_max, >= 0;
24
25 minimize objective: c_max;
26
27 s.t. r1{k in M, l in 1..n[k]}: sum{i in operations
28 s.t. r2{i in N}: sum{l in 1..n[m[i]]} y[i, l] = 1;
29 s.t. r3{k in M, l in 1..n[k]}: T[k, l] = sum{i in
30 s.t. r4{(i, j) in A, li in 1..n[m[i]], lj in 1..n[
31 s.t. r5{k in M}: t[k, 1] = s0[k];
32 s.t. r6{k in M, r in 2..n[k]}: t[k, r] = s0[k] + s
33 s.t. r7{k in M}: c_max >= t[k, n[k]] + T[k, n[k]];

```

processing data.
processing commands.
Executing on prod-exec-1.neos-server.org

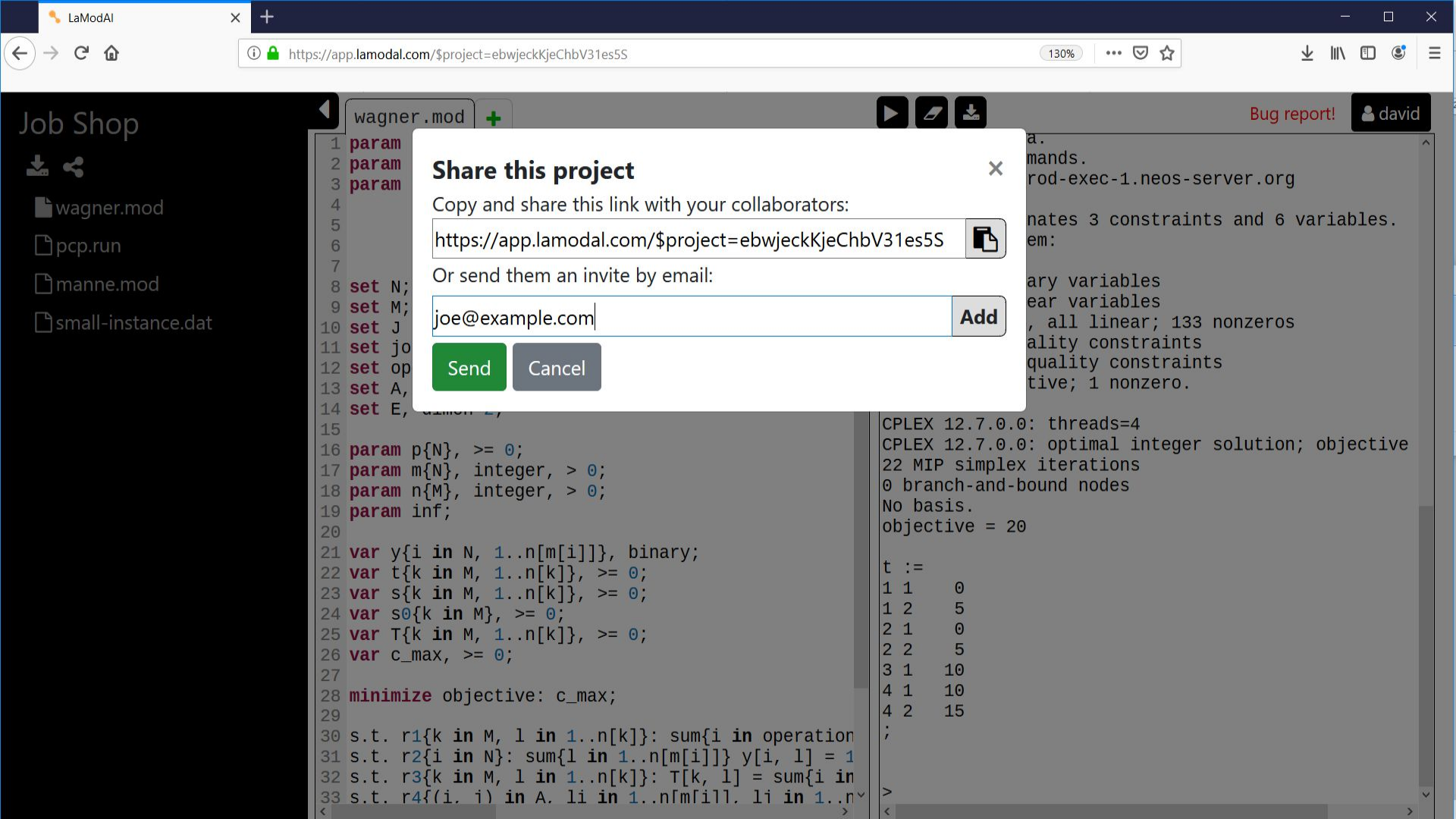
```

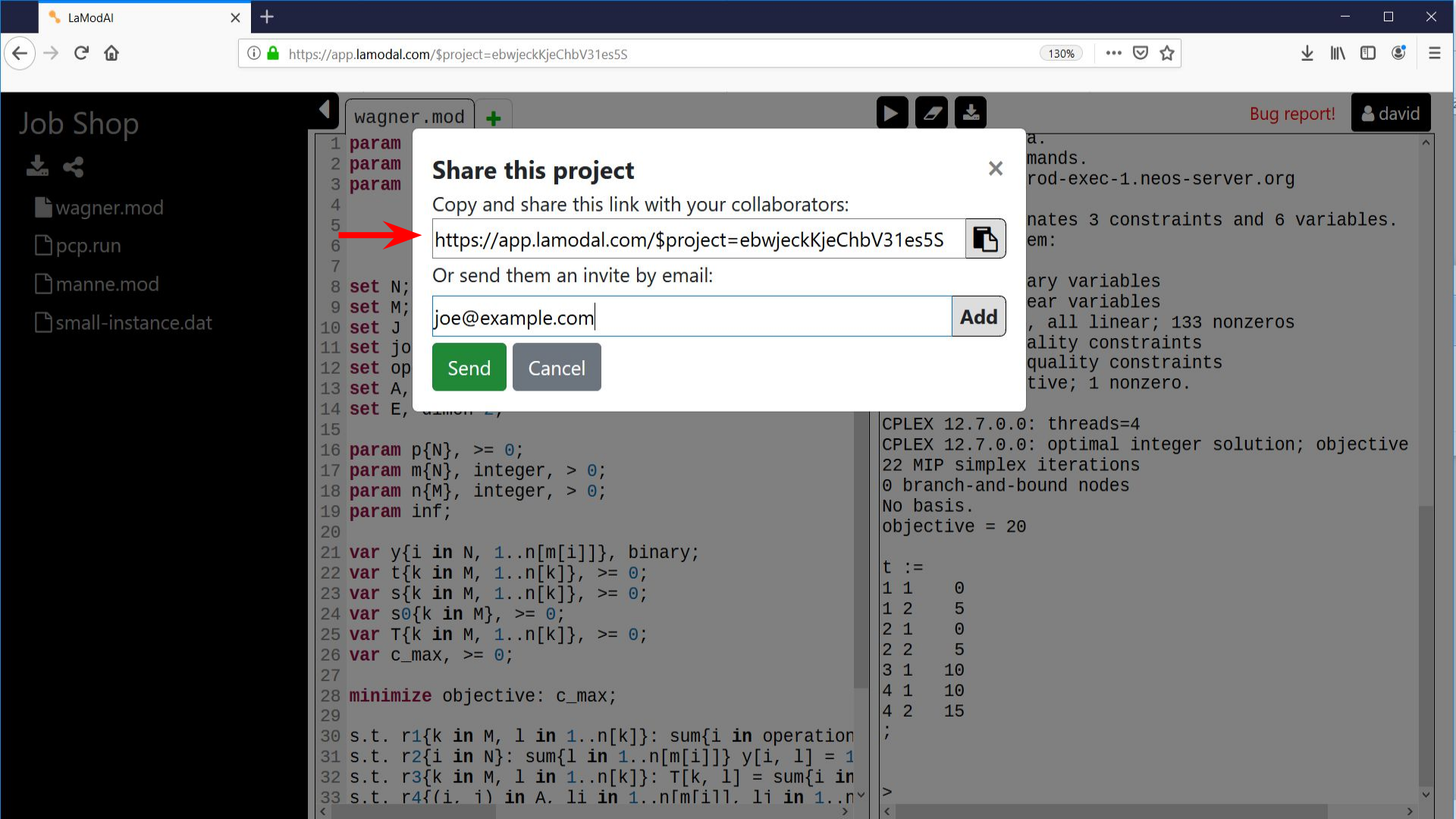
Presolve eliminates 3 constraints and 6 variables.
Adjusted problem:
33 variables:
    12 binary variables
    21 linear variables
45 constraints, all linear; 133 nonzeros
    25 equality constraints
    20 inequality constraints
1 linear objective; 1 nonzero.

CPLEX 12.7.0.0: threads=4
CPLEX 12.7.0.0: optimal integer solution; objective 2
22 MIP simplex iterations
0 branch-and-bound nodes
No basis.
objective = 20

t :=
1 1    0
1 2    5
2 1    0
2 2    5
3 1   10
4 1   10
4 2   15
;
> |

```





Share this project

Copy and share this link with your collaborators:

[https://app.lamodal.com/\\$project=ebwjeckKjeChbV31es5S](https://app.lamodal.com/$project=ebwjeckKjeChbV31es5S)

Or send them an invite by email:

Add

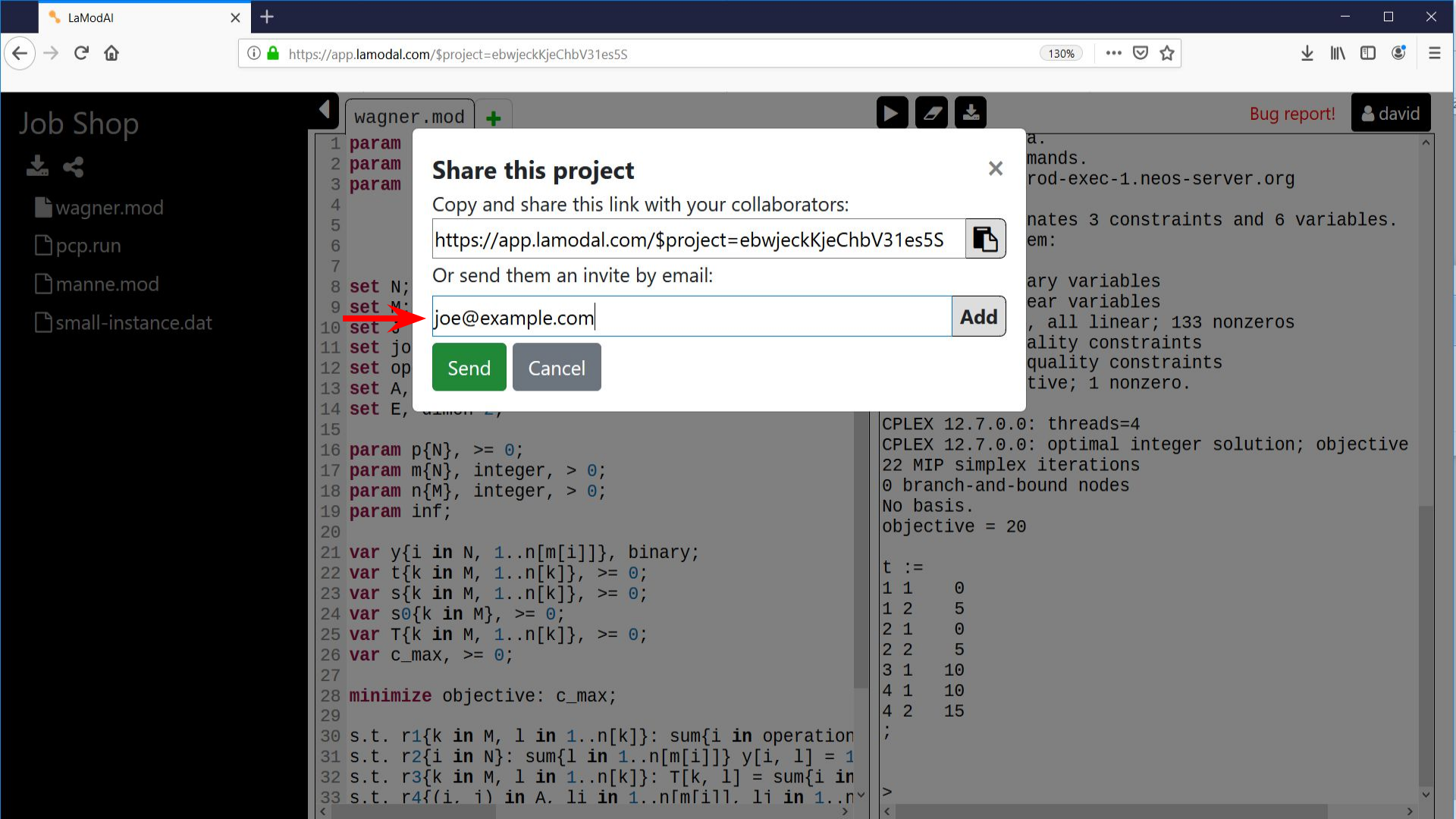
Send

Cancel

```
1 param
2 param
3 param
4
5
6
7
8 set N;
9 set M;
10 set J
11 set jo
12 set op
13 set A,
14 set E,
15
16 param p{N}, >= 0;
17 param m{N}, integer, > 0;
18 param n{M}, integer, > 0;
19 param inf;
20
21 var y{i in N, 1..n[m[i]]}, binary;
22 var t{k in M, 1..n[k]}, >= 0;
23 var s{k in M, 1..n[k]}, >= 0;
24 var s0{k in M}, >= 0;
25 var T{k in M, 1..n[k]}, >= 0;
26 var c_max, >= 0;
27
28 minimize objective: c_max;
29
30 s.t. r1{k in M, l in 1..n[k]}: sum{i in operation
31 s.t. r2{i in N}: sum{l in 1..n[m[i]]} y[i, l] = 1
32 s.t. r3{k in M, l in 1..n[k]}: T[k, l] = sum{i in
33 s.t. r4{(i, i) in A. li in 1..n[m[il]. li in 1..n
```

```
CPLEX 12.7.0.0: threads=4
CPLEX 12.7.0.0: optimal integer solution; objective
22 MIP simplex iterations
0 branch-and-bound nodes
No basis.
objective = 20
```

```
t :=
1 1    0
1 2    5
2 1    0
2 2    5
3 1   10
4 1   10
4 2   15
```



Share this project

Copy and share this link with your collaborators:

Or send them an invite by email:

```
1 param
2 param
3 param
4
5
6
7
8 set N;
9 set M;
10 set J;
11 set jo
12 set op
13 set A,
14 set E,
15
16 param p{N}, >= 0;
17 param m{N}, integer, > 0;
18 param n{M}, integer, > 0;
19 param inf;
20
21 var y{i in N, 1..n[m[i]]}, binary;
22 var t{k in M, 1..n[k]}, >= 0;
23 var s{k in M, 1..n[k]}, >= 0;
24 var s0{k in M}, >= 0;
25 var T{k in M, 1..n[k]}, >= 0;
26 var c_max, >= 0;
27
28 minimize objective: c_max;
29
30 s.t. r1{k in M, l in 1..n[k]}: sum{i in operation
31 s.t. r2{i in N}: sum{l in 1..n[m[i]]} y[i, l] = 1
32 s.t. r3{k in M, l in 1..n[k]}: T[k, l] = sum{i in
33 s.t. r4{(i, i) in A. li in 1..n[m[i]], li in 1..n
```

```
CPLEX 12.7.0.0: threads=4
CPLEX 12.7.0.0: optimal integer solution; objective
22 MIP simplex iterations
0 branch-and-bound nodes
No basis.
objective = 20

t :=
1 1    0
1 2    5
2 1    0
2 2    5
3 1   10
4 1   10
4 2   15
;
```

Job Shop

- wagner.mod
- pcp.run
- manne.mod
- small-instance.dat

Bug report! david

Job Shop

- wagner.mod
- pcp.run
- manne.mod
- small-instance.dat

```
wagner.mod +
1 param n_jobs, integer, > 0;
2 param n_operations, integer, > 0;
3 param n_machines, integer, > 0;
4
5
6
7
8 set N;
9 set M;
10 set J := 1..n_jobs;
11 set job_operations{J};
12 set operations_at{M};
13 set A, dimen 2;
14 set E, dimen 2;
15
16 param n{N} >= 0;
```

Bug report! nidoro

```
@ branch-and-bound nodes
No basis.
objective = 20

t :=
1 1 0
1 2 5
2 1 0
2 2 5
3 1 10
4 1 10
4 2 15
;

> |
```

Job Shop

- wagner.mod
- pcp.run
- manne.mod
- small-instance.dat

```
wagner.mod +
1 param n_jobs, integer, > 0;
2 param n_operations, integer, > 0;
3 param n_machines, integer, > 0;
4
5
6
7
8 set N;
9 set M;
10 set J := 1..n_jobs;
11 set job_operations{J};
12 set operations_at{M};
13 set A, dimen 2;
14 set E, dimen 2;
15
16 param n{N} >= 0;
```

Bug report! david

```
No basis.
objective = 20

t :=
1 1 0
1 2 5
2 1 0
2 2 5
3 1 10
4 1 10
4 2 15
;

>
```

LaModAI

https://app.lamodal.com/\$project=ebwjeckKjeChbV31es5S

Job Shop

- wagner.mod
- pcp.run
- manne.mod
- small-instance.dat

```
wagner.mod
1 param n_jobs, integer, > 0;
2 param n_operations, integer, > 0;
3 param n_machines, integer, > 0;
4
5 #
6 #
7
8 set N;
9 set M;
10 set J := 1..n_jobs;
11 set job_operations{J};
12 set operations_at{M};
13 set A, dimen 2;
14 set E, dimen 2;
15
16 param n{N} >= 0;
```

0 branch-and-bound nodes
No basis.
objective = 20

t :=		
1	1	0
1	2	5
2	1	0
2	2	5
3	1	10
4	1	10
4	2	15
;		

> |

Bug report! nidoro

Job Shop

- wagner.mod
- pcp.run
- manne.mod
- small-instance.dat

```
wagner.mod
1 param n_jobs, integer, > 0;
2 param n_operations, integer, > 0;
3 param n_machines, integer, > 0;
4
5 #
6 #
7
8 set N;
9 set M;
10 set J := 1..n_jobs;
11 set job_operations{J};
12 set operations_at{M};
13 set A, dimen 2;
14 set E, dimen 2;
15
16 param n{N} >= 0;
```

No basis.
objective = 20

t :=		
1	1	0
1	2	5
2	1	0
2	2	5
3	1	10
4	1	10
4	2	15
;		

>

Bug report! david

Job Shop

- wagner.mod
- pcp.run
- manne.mod
- small-instance.dat

```
wagner.mod +
1 param n_jobs, integer, > 0;
2 param n_operations, integer, > 0;
3 param n_machines, integer, > 0;
4
5 # Co
6 # S
7
8 set N;
9 set M;
10 set J := 1..n_jobs;
11 set job_operations{J};
12 set operations_at{M};
13 set A, dimen 2;
14 set E, dimen 2;
15
16 param n{N} >= 0;
```

Bug report! nidoro

```
@ branch-and-bound nodes
No basis.
objective = 20

t :=
1 1 0
1 2 5
2 1 0
2 2 5
3 1 10
4 1 10
4 2 15
;
```

Job Shop

- wagner.mod
- pcp.run
- manne.mod
- small-instance.dat

```
wagner.mod +
1 param n_jobs, integer, > 0;
2 param n_operations, integer, > 0;
3 param n_machines, integer, > 0;
4
5 # Co
6 # S
7
8 set N;
9 set M;
10 set J := 1..n_jobs;
11 set job_operations{J};
12 set operations_at{M};
13 set A, dimen 2;
14 set E, dimen 2;
```

Bug report! david

```
No basis.
objective = 20

t :=
1 1 0
1 2 5
2 1 0
2 2 5
3 1 10
4 1 10
4 2 15
;
```

Job Shop

- wagner.mod
- pcp.run
- manne.mod
- small-instance.dat

```
wagner.mod +  
1 param n_jobs, integer, > 0;  
2 param n_operations, integer, > 0;  
3 param n_machines, integer, > 0;  
4  
5 # Collab  
6 # So p  
7  
8 set N;  
9 set M;  
10 set J := 1..n_jobs;  
11 set job_operations{J};  
12 set operations_at{M};  
13 set A, dimen 2;  
14 set E, dimen 2;  
15  
16 param n{N} >= 0;
```

```
0 branch-and-bound nodes  
No basis.  
objective = 20  
  
t :=  
1 1 0  
1 2 5  
2 1 0  
2 2 5  
3 1 10  
4 1 10  
4 2 15  
;  
  
> |
```

Job Shop

- wagner.mod
- pcp.run
- manne.mod
- small-instance.dat

```
wagner.mod +  
1 param n_jobs, integer, > 0;  
2 param n_operations, integer, > 0;  
3 param n_machines, integer, > 0;  
4  
5 # Collab  
6 # So p  
7  
8 set N;  
9 set M;  
10 set J := 1..n_jobs;  
11 set job_operations{J};  
12 set operations_at{M};  
13 set A, dimen 2;  
14 set E, dimen 2;  
15  
16 param n{N} >= 0;
```

```
0 branch-and-bound nodes  
No basis.  
objective = 20  
  
t :=  
1 1 0  
1 2 5  
2 1 0  
2 2 5  
3 1 10  
4 1 10  
4 2 15  
;  
  
> |
```

Job Shop

- wagner.mod
- pcp.run
- manne.mod
- small-instance.dat

```
wagner.mod +  
1 param n_jobs, integer, > 0;  
2 param n_operations, integer, > 0;  
3 param n_machines, integer, > 0;  
4  
5 # Collaborat  
6 # So produ  
7  
8 set N;  
9 set M;  
10 set J := 1..n_jobs;  
11 set job_operations{J};  
12 set operations_at{M};  
13 set A, dimen 2;  
14 set E, dimen 2;  
15  
16 param n{N} >= 0;
```

```
0 branch-and-bound nodes  
No basis.  
objective = 20  
  
t :=  
1 1 0  
1 2 5  
2 1 0  
2 2 5  
3 1 10  
4 1 10  
4 2 15  
;  
  
> |
```

Job Shop

- wagner.mod
- pcp.run
- manne.mod
- small-instance.dat

```
wagner.mod +  
1 param n_jobs, integer, > 0;  
2 param n_operations, integer, > 0;  
3 param n_machines, integer, > 0;  
4  
5 # Collaborat  
6 # So produ  
7  
8 set N;  
9 set M;  
10 set J := 1..n_jobs;  
11 set job_operations{J};  
12 set operations_at{M};  
13 set A, dimen 2;  
14 set E, dimen 2;  
15  
16 param n{N} >= 0;
```

```
0 branch-and-bound nodes  
No basis.  
objective = 20  
  
t :=  
1 1 0  
1 2 5  
2 1 0  
2 2 5  
3 1 10  
4 1 10  
4 2 15  
;  
  
> |
```

Job Shop

- wagner.mod
- pcp.run
- manne.mod
- small-instance.dat

```
wagner.mod +  
1 param n_jobs, integer, > 0;  
2 param n_operations, integer, > 0;  
3 param n_machines, integer, > 0;  
4  
5 # Collaborating  
6 # So productiv  
7  
8 set N;  
9 set M;  
10 set J := 1..n_jobs;  
11 set job_operations{J};  
12 set operations_at{M};  
13 set A, dimen 2;  
14 set E, dimen 2;  
15  
16 param n{N} >= 0;
```

```
0 branch-and-bound nodes  
No basis.  
objective = 20  
  
t :=  
1 1 0  
1 2 5  
2 1 0  
2 2 5  
3 1 10  
4 1 10  
4 2 15  
;  
> |
```

Job Shop

- wagner.mod
- pcp.run
- manne.mod
- small-instance.dat

```
wagner.mod +  
1 param n_jobs, integer, > 0;  
2 param n_operations, integer, > 0;  
3 param n_machines, integer, > 0;  
4  
5 # Collaborating  
6 # So productiv  
7  
8 set N;  
9 set M;  
10 set J := 1..n_jobs;  
11 set job_operations{J};  
12 set operations_at{M};  
13 set A, dimen 2;  
14 set E, dimen 2;  
15  
16 param n{N} >= 0;
```

```
0 branch-and-bound nodes  
No basis.  
objective = 20  
  
t :=  
1 1 0  
1 2 5  
2 1 0  
2 2 5  
3 1 10  
4 1 10  
4 2 15  
;  
> |
```

Job Shop

- wagner.mod
- pcp.run
- manne.mod
- small-instance.dat

```
wagner.mod +  
1 param n_jobs, integer, > 0;  
2 param n_operations, integer, > 0;  
3 param n_machines, integer, > 0;  
4  
5 # Collaborating is f  
6 # So productive!  
7  
8 set N;  
9 set M;  
10 set J := 1..n_jobs;  
11 set job_operations{J};  
12 set operations_at{M};  
13 set A, dimen 2;  
14 set E, dimen 2;  
15  
16 param n{N} >= 0;
```

```
0 branch-and-bound nodes  
No basis.  
objective = 20  
  
t :=  
1 1 0  
1 2 5  
2 1 0  
2 2 5  
3 1 10  
4 1 10  
4 2 15  
;  
> |
```

Job Shop

- wagner.mod
- pcp.run
- manne.mod
- small-instance.dat

```
wagner.mod +  
1 param n_jobs, integer, > 0;  
2 param n_operations, integer, > 0;  
3 param n_machines, integer, > 0;  
4  
5 # Collaborating is f  
6 # So productive!  
7  
8 set N;  
9 set M;  
10 set J := 1..n_jobs;  
11 set job_operations{J};  
12 set operations_at{M};  
13 set A, dimen 2;  
14 set E, dimen 2;  
15  
16 param n{N} >= 0;
```

```
No basis.  
objective = 20  
  
t :=  
1 1 0  
1 2 5  
2 1 0  
2 2 5  
3 1 10  
4 1 10  
4 2 15  
;  
>
```

Job Shop

- wagner.mod
- pcp.run
- manne.mod
- small-instance.dat

```
wagner.mod +
1 param n_jobs, integer, > 0;
2 param n_operations, integer, > 0;
3 param n_machines, integer, > 0;
4
5 # Collaborating is fun
6 # So productive!
7
8 set N;
9 set M;
10 set J := 1..n_jobs;
11 set job_operations{J};
12 set operations_at{M};
13 set A, dimen 2;
14 set E, dimen 2;
15
16 param n{N} >= 0;
```

```
0 branch-and-bound nodes
No basis.
objective = 20

t :=
1 1 0
1 2 5
2 1 0
2 2 5
3 1 10
4 1 10
4 2 15
;
```

Job Shop

- wagner.mod
- pcp.run
- manne.mod
- small-instance.dat

```
wagner.mod +
1 param n_jobs, integer, > 0;
2 param n_operations, integer, > 0;
3 param n_machines, integer, > 0;
4
5 # Collaborating is fun
6 # So productive!
7
8 set N;
9 set M;
10 set J := 1..n_jobs;
11 set job_operations{J};
12 set operations_at{M};
13 set A, dimen 2;
14 set E, dimen 2;
```

```
No basis.
objective = 20

t :=
1 1 0
1 2 5
2 1 0
2 2 5
3 1 10
4 1 10
4 2 15
;
```


Job Shop

- wagner.mod
- pcp.run
- manne.mod
- small-instance.dat

```
wagner.mod +  
1 param n_jobs, integer, > 0;  
2 param n_operations, integer, > 0;  
3 param n_machines, integer, > 0;  
4  
5 # Collaborating is fun!  
6 # So productive! Wow!  
7  
8 set N;  
9 set M;  
10 set J := 1..n_jobs;  
11 set job_operations{J};  
12 set operations_at{M};  
13 set A, dimen 2;  
14 set E, dimen 2;  
15  
16 param n{N} >= 0;
```

```
0 branch-and-bound nodes  
No basis.  
objective = 20  
  
t :=  
1 1 0  
1 2 5  
2 1 0  
2 2 5  
3 1 10  
4 1 10  
4 2 15  
;  
> |
```

Job Shop

- wagner.mod
- pcp.run
- manne.mod
- small-instance.dat

```
wagner.mod +  
1 param n_jobs, integer, > 0;  
2 param n_operations, integer, > 0;  
3 param n_machines, integer, > 0;  
4  
5 # Collaborating is fun!  
6 # So productive! Wow!  
7  
8 set N;  
9 set M;  
10 set J := 1..n_jobs;  
11 set job_operations{J};  
12 set operations_at{M};  
13 set A, dimen 2;  
14 set E, dimen 2;  
15  
16 param n{N} >= 0;
```

```
0 branch-and-bound nodes  
No basis.  
objective = 20  
  
t :=  
1 1 0  
1 2 5  
2 1 0  
2 2 5  
3 1 10  
4 1 10  
4 2 15  
;  
> |
```

Job Shop

- wagner.mod
- pcp.run
- manne.mod
- small-instance.dat

```
wagner.mod +  
1 param n_jobs, integer, > 0;  
2 param n_operations, integer, > 0;  
3 param n_machines, integer, > 0;  
4  
5 # Collaborating is fun!  
6 # So productive! Wow!  
7  
8 set N;  
9 set M;  
10 set J := 1..n_jobs;  
11 set job_operations{J};  
12 set operations_at{M};  
13 set A, dimen 2;  
14 set E, dimen 2;  
15  
16 param n{N} >= 0;
```

```
0 branch-and-bound nodes  
No basis.  
objective = 20  
  
t :=  
1 1 0  
1 2 5  
2 1 0  
2 2 5  
3 1 10  
4 1 10  
4 2 15  
;  
> |
```

Job Shop

- wagner.mod
- pcp.run
- manne.mod
- small-instance.dat

```
wagner.mod +  
1 param n_jobs, integer, > 0;  
2 param n_operations, integer, > 0;  
3 param n_machines, integer, > 0;  
4  
5 # Collaborating is fun!  
6 # So productive! Wow!  
7  
8 set N;  
9 set M;  
10 set J := 1..n_jobs;  
11 set job_operations{J};  
12 set operations_at{M};  
13 set A, dimen 2;  
14 set E, dimen 2;  
15  
16 param n{N} >= 0;
```

```
0 branch-and-bound nodes  
No basis.  
objective = 20  
  
t :=  
1 1 0  
1 2 5  
2 1 0  
2 2 5  
3 1 10  
4 1 10  
4 2 15  
;  
> |
```

Job Shop

- wagner.mod
- pcp.run
- manne.mod
- small-instance.dat

```
wagner.mod +  
1 param n_jobs, integer, > 0;  
2 param n_operations, integer, > 0;  
3 param n_machines, integer, > 0;  
4  
5 # Collaborating is fun!  
6 # So productive! Wow!  
7  
8 set N;  
9 set M;  
10 set J := 1..n_jobs;  
11 set job_operations{J};  
12 set operations_at{M};  
13 set A, dimen 2;  
14 set E, dimen 2;  
15  
16 param n{N} >= 0;
```







```
0 branch-and-bound nodes  
No basis.  
objective = 20  
  
t :=  
1 1 0  
1 2 5  
2 1 0  
2 2 5  
3 1 10  
4 1 10  
4 2 15  
;  
> |
```

Job Shop

- wagner.mod
- pcp.run
- manne.mod
- small-instance.dat

```
wagner.mod +  
1 param n_jobs, integer, > 0;  
2 param n_operations, integer, > 0;  
3 param n_machines, integer, > 0;  
4  
5 # Collaborating is fun!  
6 # So productive! Wow!  
7  
8 set N;  
9 set M;  
10 set J := 1..n_jobs;  
11 set job_operations{J};  
12 set operations_at{M};  
13 set A, dimen 2;  
14 set E, dimen 2;  
15  
16 param n{N} >= 0;
```

```
0 branch-and-bound nodes  
No basis.  
objective = 20  
  
t :=  
1 1 0  
1 2 5  
2 1 0  
2 2 5  
3 1 10  
4 1 10  
4 2 15  
;  
> |
```

-  
-  wagner.mod
-  pcp.run
-  manne.mod
-  small-instance.dat

wagner.mod +

```

1 param n_jobs, integer, > 0;
2 param n_operations, integer, > 0;
3 param n_machines, integer, > 0;
4
5 set N;
6 set M;
7 set J := 1..n_jobs;
8 set job_operations{J};
9 set operations_at{M};
10 set A, dimen 2;
11 set E, dimen 2;
12
13 param p{N}, >= 0;
14 param m{N}, integer, > 0;
15 param n{M}, integer, > 0;
16 param inf;
17
18 var y{i in N, 1..n[m[i]]}, binary;
19 var t{k in M, 1..n[k]}, >= 0;
20 var s{k in M, 1..n[k]}, >= 0;
21 var s0{k in M}, >= 0;
22 var T{k in M, 1..n[k]}, >= 0;
23 var c_max, >= 0;
24
25 minimize objective: c_max;
26
27 s.t. r1{k in M, l in 1..n[k]}: sum{i in operations
28 s.t. r2{i in N}: sum{l in 1..n[m[i]]} y[i, l] = 1;
29 s.t. r3{k in M, l in 1..n[k]}: T[k, l] = sum{i in
30 s.t. r4{(i, j) in A, li in 1..n[m[i]], lj in 1..n[
31 s.t. r5{k in M}: t[k, 1] = s0[k];
32 s.t. r6{k in M, r in 2..n[k]}: t[k, r] = s0[k] + s
33 s.t. r7{k in M}: c_max >= t[k, n[k]] + T[k, n[k]];

```

```

processing data.
processing commands.
Executing on prod-exec-1.neos-server.org

Presolve eliminates 3 constraints and 6 variables.
Adjusted problem:
33 variables:
    12 binary variables
    21 linear variables
45 constraints, all linear; 133 nonzeros
    25 equality constraints
    20 inequality constraints
1 linear objective; 1 nonzero.

CPLEX 12.7.0.0: threads=4
CPLEX 12.7.0.0: optimal integer solution; objective 2
22 MIP simplex iterations
0 branch-and-bound nodes
No basis.
objective = 20

t :=
1 1    0
1 2    5
2 1    0
2 2    5
3 1   10
4 1   10
4 2   15
;
> |

```

- wagner.mod
- pcp.run
- manne.mod
- small-instance.dat

```

1 param n_jobs, integer, > 0;
2 param n_operations, integer, > 0;
3 param n_machines, integer, > 0;
4
5 set N;
6 set M;
7 set J := 1..n_jobs;
8 set job_operations{J};
9 set operations_at{M};
10 set A, dimen 2;
11 set E, dimen 2;
12
13 param p{N}, >= 0;
14 param m{N}, integer, > 0;
15 param n{M}, integer, > 0;
16 param inf;
17
18 var y{i in N, 1..n[m[i]]}, binary;
19 var t{k in M, 1..n[k]}, >= 0;
20 var s{k in M, 1..n[k]}, >= 0;
21 var s0{k in M}, >= 0;
22 var T{k in M, 1..n[k]}, >= 0;
23 var c_max, >= 0;
24
25 minimize objective: c_max;
26
27 s.t. r1{k in M, l in 1..n[k]}: sum{i in operations
28 s.t. r2{i in N}: sum{l in 1..n[m[i]]} y[i, l] = 1;
29 s.t. r3{k in M, l in 1..n[k]}: T[k, l] = sum{i in
30 s.t. r4{(i, j) in A, li in 1..n[m[i]], lj in 1..n[
31 s.t. r5{k in M}: t[k, 1] = s0[k];
32 s.t. r6{k in M, r in 2..n[k]}: t[k, r] = s0[k] + s
33 s.t. r7{k in M}: c_max >= t[k, n[k]] + T[k, n[k]];

```

```

processing data.
processing commands.
Executing on prod-exec-1.neos-server.org







Presolve eliminates 3 constraints and 6 variables.
Adjusted problem:
33 variables:
    12 binary variables
    21 linear variables
45 constraints, all linear; 133 nonzeros
    25 equality constraints
    20 inequality constraints
1 linear objective; 1 nonzero.

CPLEX 12.7.0.0: threads=4
CPLEX 12.7.0.0: optimal integer solution; objective 2
22 MIP simplex iterations
0 branch-and-bound nodes
No basis.
objective = 20

t :=
1 1    0
1 2    5
2 1    0
2 2    5
3 1   10
4 1   10
4 2   15
;

> |neos submit wagner.mod small-instance.dat pcp.run

```

-  
-  wagner.mod
-  pcp.run
-  manne.mod
-  small-instance.dat

wagner.mod +

```

1 param n_jobs, integer, > 0;
2 param n_operations, integer, > 0;
3 param n_machines, integer, > 0;
4
5 set N;
6 set M;
7 set J := 1..n_jobs;
8 set job_operations{J};
9 set operations_at{M};
10 set A, dimen 2;
11 set E, dimen 2;
12
13 param p{N}, >= 0;
14 param m{N}, integer, > 0;
15 param n{M}, integer, > 0;
16 param inf;
17
18 var y{i in N, 1..n[m[i]]}, binary;
19 var t{k in M, 1..n[k]}, >= 0;
20 var s{k in M, 1..n[k]}, >= 0;
21 var s0{k in M}, >= 0;
22 var T{k in M, 1..n[k]}, >= 0;
23 var c_max, >= 0;
24
25 minimize objective: c_max;
26
27 s.t. r1{k in M, l in 1..n[k]}: sum{i in operations
28 s.t. r2{i in N}: sum{l in 1..n[m[i]]} y[i, l] = 1;
29 s.t. r3{k in M, l in 1..n[k]}: T[k, l] = sum{i in
30 s.t. r4{(i, j) in A, li in 1..n[m[i]], lj in 1..n[
31 s.t. r5{k in M}: t[k, 1] = s0[k];
32 s.t. r6{k in M, r in 2..n[k]}: t[k, r] = s0[k] + s
33 s.t. r7{k in M}: c_max >= t[k, n[k]] + T[k, n[k]];

```

processing data.
processing commands.
Executing on prod-exec-1.neos-server.org

```

Presolve eliminates 3 constraints and 6 variables.
Adjusted problem:
33 variables:
    12 binary variables
    21 linear variables
45 constraints, all linear; 133 nonzeros
    25 equality constraints
    20 inequality constraints
1 linear objective; 1 nonzero.

CPLEX 12.7.0.0: threads=4
CPLEX 12.7.0.0: optimal integer solution; objective 2
22 MIP simplex iterations
0 branch-and-bound nodes
No basis.
objective = 20

t :=
1 1    0
1 2    5
2 1    0
2 2    5
3 1   10
4 1   10
4 2   15
;

> |neos submit wagner.mod small-instance.dat pcp.run

```




Student

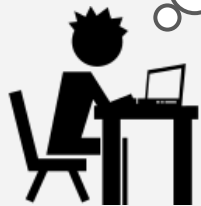


Student



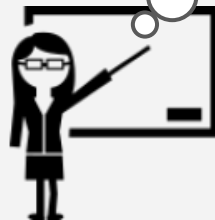
Teacher

Tasks



Student

Tasks



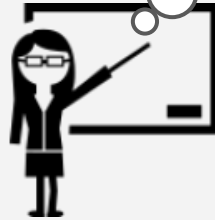
Teacher

**Tasks
Pains**



Student

**Tasks
Pains**



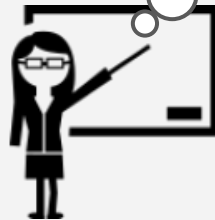
Teacher

Tasks
Pains
Gains



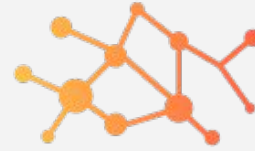
Student

Tasks
Pains
Gains



Teacher

Student
Tasks



LaModAI
Services

Pains

Pain Relievers

Gains

Gain Creators

Student

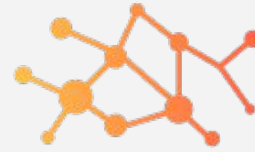


Tasks

Individual and group assignments

Pains

Gains



LaModAI

Services

Pain Relievers

Gain Creators

Student



Tasks

Individual and group assignments

Pains

Gains



LaModAI

Services

IDE with support for live collaboration

Pain Relievers

Gain Creators

Student



Tasks

Individual and group assignments

Pains

Interpreter/Solver download and setup

Gains



LaModAI

Services

IDE with support for live collaboration

Pain Relievers

Gain Creators

Student



Tasks

Individual and group assignments

Pains

Interpreter/Solver download and setup

Gains



LaModAI

Services

IDE with support for live collaboration

Pain Relievers

Cloud-based storage and remote execution

Gain Creators

Student



Tasks

Individual and group assignments

Pains

Interpreter/Solver download and setup

Gains

Learn mathematical modeling



LaModAI

Services

IDE with support for live collaboration

Pain Relievers

Cloud-based storage and remote execution

Gain Creators

Student Tasks



Individual and group assignments

Pains

Interpreter/Solver download and setup

Gains

Learn mathematical modeling



LaModAI Services

IDE with support for live collaboration

Pain Relievers

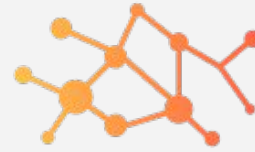
Cloud-based storage and remote execution

Gain Creators

Better way to ask professor's feedback

Teacher

Tasks



LaModAI

Services

Pains

Pain Relievers

Gains

Gain Creators

Teacher

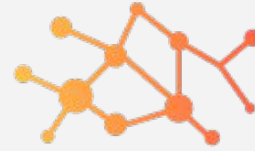


Tasks

Teach mathematical modeling in practice

Pains

Gains



**LaModAI
Services**

Pain Relievers

Gain Creators

Teacher



Tasks

Teach mathematical modeling in practice

Pains

Gains



LaModAI

Services

Can be used for both individual and group work

Pain Relievers

Gain Creators

Teacher



Tasks

Teach mathematical modeling in practice

Pains

Teach how to setup

Gains



LaModAI

Services

Can be used for both individual and group work

Pain Relievers

Gain Creators

Teacher



Tasks

Teach mathematical modeling in practice

Pains

Teach how to setup

Gains



LaModAI

Services

Can be used for both individual and group work

Pain Relievers

No setup required

Gain Creators

Teacher



Tasks

Teach mathematical modeling in practice

Pains

Teach setup, provide feedback and check correctness of assignments

Gains



LaModAI

Services

Can be used for both individual and group work

Pain Relievers

No setup required

Gain Creators

Teacher



Tasks

Teach mathematical modeling in practice

Pains

Teach setup, provide feedback and check correctness of assignments

Gains



LaModAI

Services

Can be used for both individual and group work

Pain Relievers

No setup and remote access to students' assignments

Gain Creators

Teacher



Tasks

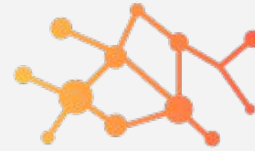
Teach mathematical modeling in practice

Pains

Teach setup, provide feedback and check correctness of assignments

Gains

Knowledgeable students as soon as possible



LaModAI

Services

Can be used for both individual and group work

Pain Relievers

No setup and remote access to students' assignments

Gain Creators

Teacher



Tasks

Teach mathematical modeling in practice

Pains

Teach setup, provide feedback and check correctness of assignments

Gains

Knowledgeable students as soon as possible



LaModAI

Services

Can be used for both individual and group work

Pain Relievers

No setup and remote access to students' assignments

Gain Creators

Set of features that accelerate the acquisition of modeling knowledge



lamodal.com - What's next?

sample projects and templates

video tutorials

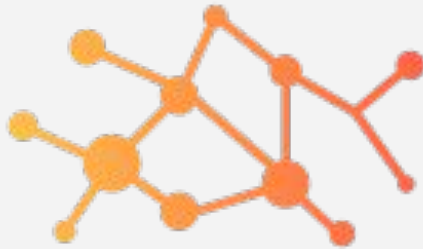
virtual classrooms

automatic assignment correctness check

project versioning

GitHub integration

private servers



LaModAI

Thank you!

Davi Doro

davi_doro@hotmail.com

Ricardo Camargo

rcamargo@dep.ufmg.br

