Jeffrey Kantor <jeff.kantor@gmail.com>

[Help-glpk] Ordered bin packing

11 messages

Dmitri Goutnik <dg@syrec.org> Sun, Jan 27, 2013 at 1:34 PM
To: help-glpk@gnu.org

Hello everyone,

I modified bpp.mod to pack items in predefined number of bins of unlimited capacity so that weight is (approximately) evenly distributed between bins. In addition to weight, each item has an "order" or "position". Is there a way to write an additional model constraint so that items are preferably packed in compact order, e.g. items with sequential positions should go to the the same container? For example, for this solution:

Bin 1: total weight 110
item 1 (40)
item 4 (70) <--

Bin 2: total weight 90
item 5 (50)
item 6 (40)

Bin 3: total weight 130
item 2 (60) -->
item 3 (30)
item 7 (15)
item 8 (25)

is there a way to add such constraint so item 2 would preferably be packed to bin 1?

Thanks,
Dmitri

Help-glpk mailing list
Help-glpk@gnu.org
https://lists.gnu.org/mailman/listinfo/help-glpk

Jeffrey Kantor <Kantor.1@nd.edu> Sun, Jan 27, 2013 at 1:40 PM
To: Dmitri Goutnik <dg@syrec.org>
Cc: GLPK <help-glpk@gnu.org>

Hi Dmitri,

This could be done through the objective function. But before going there, I'm wondering a bit
about the solution you're showing for your example. Wouldn't you want move item 7 from Bin 3 to Bin 2 to meet your primary objective?

Jeff
[Quoted text hidden]

Dmitri Goutnik <dg@syrec.org> Sun, Jan 27, 2013 at 1:55 PM
To: Jeffrey Kantor <Kantor.1@nd.edu>
Cc: GLPK <help-glpk@gnu.org>

Hi Jeff,

Thanks, yes that would be preferable packing. My goal is to pack items as "tightly" as possible while preserving even (within some predefined margin) weight distribution between bins.

Best regards,
Dmitri

[Quoted text hidden]

Jeffrey Kantor <Kantor.1@nd.edu> Sun, Jan 27, 2013 at 2:16 PM
To: Dmitri Goutnik <dg@syrec.org>

Dmitri,

This is a problem with two objectives which, depending on problem data, may conflict with each other. So the best we can do is to find tradeoffs between the conflicting objectives. Here's one solution where alpha is a parameter expressing the relative significance of the two objectives.

```
set BINS := 1..3;
set ITEMS := 1..8;

param weight{ITEMS};

var wmax >= 0;
var x{i in ITEMS, b in BINS} binary;

s.t. A{i in ITEMS}: sum{b in BINS} x[i,b] = 1;
s.t. B{b in BINS}: sum{i in ITEMS} weight[i]*x[i,b] <= wmax;

minimize obj: wmax + alpha*sum{b in BINS, i in ITEMS} (card(BINS)-b)*i*x[i,b];

solve;

for {b in BINS}: {
   printf "Bin %d: total weight %d\n", b, sum{i in ITEMS} weight[i]*x[i,b];
   printf {i in ITEMS : x[i,b] = 1}: "      item %d (%d)\n", i, weight[i];
   printf "\n";
}

data;

param weight :=
   1   40
   2   60
   3   30
   4   70
   5   50
   6   40
   7   15
   8   25 ;

end;
```

```
Bin 1: total weight 110
     item 1 (40)
     item 4 (70)

Bin 2: total weight 110
     item 2 (60)
     item 5 (50)

Bin 3: total weight 110
     item 3 (30)
     item 6 (40)
     item 7 (15)
     item 8 (25)
```

Another solution would be to solve the problem in two phases. The first phase determines the minimum weight for each bin (for

this data, 110), a second phase that seeks the tightest packing subject to a bound on bin weight that's greater than or equal to the minimum possible weight. Here's a solution to that problem where a bound of 130 is used:

```
set BINS := 1..3;
set ITEMS := 1..8;

param weight{ITEMS};
param wmax;

var x{i in ITEMS, b in BINS} binary;

s.t. A{i in ITEMS}: sum{b in BINS} x[i,b] = 1;
s.t. B{b in BINS}: sum{i in ITEMS} weight[i]*x[i,b] <= wmax;

minimize obj: sum{b in BINS, i in ITEMS} (card(BINS)-b)*i*x[i,b];

solve;

for {b in BINS}: {
   printf "Bin %d: total weight %d\n", b, sum{i in ITEMS} weight[i]*x[i,b];
   printf {i in ITEMS : x[i,b] = 1}: "     item %d (%d)\n", i, weight[i];
   printf "\n";
}

data;

param wmax := 130;

param weight :=
   1   40
   2   60
   3   30
   4   70
   5   50
   6   40
   7   15
   8   25 ;

end;
```

```
Bin 1: total weight 100
     item 1 (40)
     item 2 (60)

Bin 2: total weight 100
     item 3 (30)
     item 4 (70)

Bin 3: total weight 130
     item 5 (50)
     item 6 (40)
     item 7 (15)
     item 8 (25)
```

[Quoted text hidden]

Jeffrey Kantor <Kantor.1@nd.edu> Sun, Jan 27, 2013 at 2:17 PM
To: Dmitri Goutnik <dg@syrec.org>
Cc: GLPK <help-glpk@gnu.org>

[Quoted text hidden]
[Quoted text hidden]

```
Bin 1: total weight 110
     item 1 (40)
     item 4 (70)

Bin 2: total weight 110
     item 2 (60)
     item 5 (50)

Bin 3: total weight 110
     item 3 (30)
     item 6 (40)
     item 7 (15)
     item 8 (25)
```

[Quoted text hidden]

```
Bin 1: total weight 100
     item 1 (40)
     item 2 (60)

Bin 2: total weight 100
     item 3 (30)
     item 4 (70)

Bin 3: total weight 130
     item 5 (50)
     item 6 (40)
     item 7 (15)
     item 8 (25)
```

[Quoted text hidden]

Dmitri Goutnik <dg@syrec.org> Sun, Jan 27, 2013 at 2:57 PM
To: Jeffrey Kantor <Kantor.1@nd.edu>
Cc: GLPK <help-glpk@gnu.org>

Jeff,

Wow, thanks a lot! That is exactly what I was looking for.

Best regards,
Dmitri
[Quoted text hidden]

Jeffrey Kantor <Kantor.1@nd.edu> Sun, Jan 27, 2013 at 3:37 PM
To: Dmitri Goutnik <dg@syrec.org>
Cc: GLPK <help-glpk@gnu.org>

Dmitri,

One unanswered question is to equalize the bin weights as much as possible subject to tighest packing. Below is a solution. There's probably a more elegant formulation.

In any event, you now have solutions to special cases of your more general multiobjective problem.

-- minimize maximum weight of any bin
-- tightest packing subject to maximum weight on any bin
-- equalize bin weights subject to tightest packing

Jeff

```
set BINS := 1..3;
set ITEMS := 1..8;

param weight{ITEMS};
param M := 1000;

var wmax >= 0;
var wmin >= 0;
var x{i in ITEMS, b in BINS} binary;
var lmin{b in BINS};
var lmax{b in BINS};

s.t. A{i in ITEMS}: sum{b in BINS} x[i,b] = 1;
s.t. B{b in BINS}: sum{i in ITEMS} weight[i]*x[i,b] <= wmax;
s.t. C{b in BINS}: sum{i in ITEMS} weight[i]*x[i,b] >= wmin;
s.t. D{b in BINS, i in ITEMS}: lmin[b] <= i*x[i,b] + M*(1-x[i,b]);
s.t. E{b in BINS, i in ITEMS}: lmax[b] >= i*x[i,b];
s.t. F{b in BINS, c in BINS : b < c}: lmax[b] <= lmin[c];

minimize obj: wmax-wmin;

solve;

for {b in BINS}: {
    printf "Bin %d: total weight %d\n", b, sum{i in ITEMS} weight[i]*x[i,b];
    printf {i in ITEMS : x[i,b] = 1}: "     item %d (%d)\n", i, weight[i];
    printf "\n";
}

data;

param weight :=
    1   40
    2   60
    3   30
    4   70
    5   50
    6   40
    7   15
    8   25 ;

end;
```

```
Bin 1: total weight 100
     item 1 (40)
     item 2 (60)

Bin 2: total weight 100
     item 3 (30)
     item 4 (70)

Bin 3: total weight 130
     item 5 (50)
     item 6 (40)
     item 7 (15)
     item 8 (25)
```

[Quoted text hidden]

Dmitri Goutnik <dg@syrec.org> Sun, Jan 27, 2013 at 5:39 PM
To: Jeffrey Kantor <Kantor.1@nd.edu>

Jeff,

Thanks again for your help. All three solutions are more than acceptable for my problem (workload allocation).

Best regards,
Dmitri
[Quoted text hidden]

Jeffrey Kantor <Kantor.1@nd.edu> Sun, Jan 27, 2013 at 5:41 PM
To: Dmitri Goutnik <dg@syrec.org>

Dmitri,

Great. I teach a course on these topics, and am always looking for good examples. If you can, could you give me a little background so that I might use it as a teaching example? It's a nice problem.

Jeff
[Quoted text hidden]

Dmitri Goutnik <dg@syrec.org> Sun, Jan 27, 2013 at 6:03 PM
To: Jeffrey Kantor <Kantor.1@nd.edu>

Jeff,

Sure, I'm working on web application for managing hospitality services company. The company have number of employees that perform hotel room cleaning/maintenance. Each day they have to allocate available workload (measured in minutes/room depending on type of the work that needs to be done) to a number of employees, so that each employee has approximately equal shift duration and rooms should be allocated as tightly as possible (i.e. preferably they should be on the same floor and close one to another). Currently the shift manager does this allocation manually, which is probably not an easy task (in example sheet that I have they do allocation for 59 rooms on 16 floors to 9 employees).

Best regards,
Dmitri
[Quoted text hidden]

Jeffrey Kantor <Kantor.1@nd.edu> Sun, Jan 27, 2013 at 6:12 PM
To: Dmitri Goutnik <dg@syrec.org>

With that extra information becomes an excellent teaching example. Thanks for the background, and good luck with the web application.

Jeff
[Quoted text hidden]