

Alexander Tesch

Ralf Borndörfer

ConfOpt - A conference schedule optimizer

User Guide

31.10.2012

0. Contents

1. The conference planning problem.....	2
2. System requirements.....	3
3. Program execution.....	4
4. Input files.....	4
5. Output files.....	9
6. Further configurations.....	11
7. Typical problems.....	17
8. Data considered for ISMP 2012.....	18
9. Lessons learned.....	19

This program was developed to improve the schedule of **ISMP 2012** – conference. It refers to its structure of a conference schedule and helps to assign preformed sessions to rooms and dates with respect to specific constraints.

A first impression of the main problem and a short description of the “conference planning problem” is given in section one. For execution of the program you need to fulfill the system requirements which are listed in the second section. A detailed instruction on how the program is being started and where input data has to be added is given in the sections “Program execution” and “Input files”. Afterwards in the final solution you can choose between different views of the schedule, where each view has its own output file. These are explained in section 5. You can adjust more configurations like parameters and objective coefficients. The way you can do this is explained in “Further configurations”. If there are some difficulties during the preparation of the schedule a short FAQ in section 7 may help to clarify some misunderstandings. Some data fields require a good choice of values to obtain better results. To give you an idea of how these data can be obtained there are some formulas we used for ISMP 2012 in section 8. From our point of view there are some things we would do different when scheduling such a big conference again. Therefore you will find some “Lessons learned” in the correspondent section.

1. The conference planning problem

We have given a conference which lasts for a certain period. Within this period talks are given at fixed times and mostly a bunch of talks are given directly in succession. In the case of ISMP we have an aggregation of three talks which are given directly after each other. This aggregation we call a session. Consequently sessions are given at fixed timeslots at the conference. In this model we require that the set of talks belonging to a session is known in advance. That means we do not plan the assignment of talks to the sessions.

Furthermore every session belongs to exactly one cluster. Clusters are can be seen as superordinate topics where each session/talk must be assigned to. This facilitates the choice between the various sessions.

At large conferences there are much more sessions available than timeslots. Consequently sessions will take place in parallel and more than one room has to be used. Furthermore every room has a limited number of seats. In general sessions will have different attendance counts, mainly caused in personal preference, interest or popularity of the speaker. Finally more attractive sessions have to appear in bigger rooms. But in most cases the selection of big rooms is strictly restricted. Finally it is very important to have a clever allocation, such that capacities are not exceeded. This makes planning more complex.

In addition to that a lot of individual requests can occur. Some participants only can take part within a certain time-interval, two sessions must not take place at the same time or they should even appear directly after another.

One person might also speak more than once at the conference, as a speaker or chairman. So it has to be ensured that the corresponding sessions will not overlap temporally.

If possible, it is also important to have all clusters available at every time in conference. People who may have a huge interest in a specific cluster will not feel comfortable, when having a lot of parallel sessions of their cluster, but also having timeslots where no session occurs. This also holds for very popular sessions, which should be equally distributed over all time-slots to make the conference being interesting at any time.

Another problem is the distance people have to overcome when changing the rooms between the talks. Long distances lead to more walking time and more hectic rushes in the short breaks between the talks. This will make people feel less comfortable. Finally, the goal is to have talks with similar topics in nearby rooms.

In the end the general planning problem is very simple:

Find an assignment of every session to a room- and timeslot, where each room can be used only once per timeslot.

But including all the constraints mentioned above, this leads to a complex planning problem, where this program may help to find proper solutions.

2. System requirements

For the moment the program can be used on LINUX systems only.

The code is written in JAVA. Make sure you have an up-to-date version available.

<http://www.java.com>

The ZIMPL modeling language was used to create the Mixed-Integer-Programs. It is open source and is used to translate MIP and LP models to LP-files. It can be downloaded at the ZIB homepage:

<http://zimpl.zib.de/>

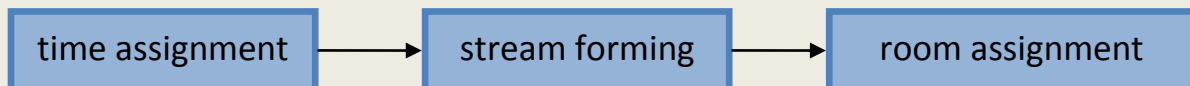
To run the program the PATH-variable must be set to the ZIMPL-folder, such that ZIMPL can be started via the “zimpl ...” -command.

For MIP-solving the commercial solver GUROBI was used. Therefore a license must be purchased:

<http://www.gurobi.com/>

3. Program execution

We used a Mixed-Integer-Programming approach to solve the master problem. Unfortunately solving the whole program leads to very high computation times, so it is divided into three sub-problems which are solved consecutively:



a. “time assignment” (MIP1)

In this model the sessions are assigned to suitable timeslots. It is solved with a MIP model. Therefore sessions of a specific cluster are partitioned to obtain a better “stream forming” result (step 2). The even distribution of clusters and attractiveness of the sessions is also done here.

b. “stream forming” (MIP2)

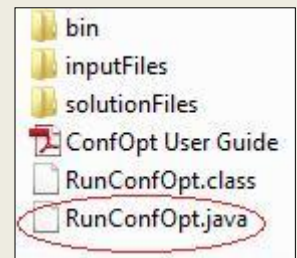
The next step is to determine which sessions should be held consecutively in the same room (*stream*). Therefore a MIP model is used, such that rooms are mainly occupied by sessions of one cluster.

c. “room assignment” (MIP3)

In the “room assignment” we assign each stream to a “real” room, such that “similar” streams are assigned to “nearby” rooms, according to similarity and distance values.

This sequence of optimization model runs automatically when executing the program:

You can find the corresponding Main-class in the ConfOpt folder. It can be started by typing “**java RunConfOpt**” into your UNIX-shell. It should work without setting the classpath, otherwise you have to set it to the ConfOpt folder.



The console output will give you additional information about the status of the computation.

4. Input files

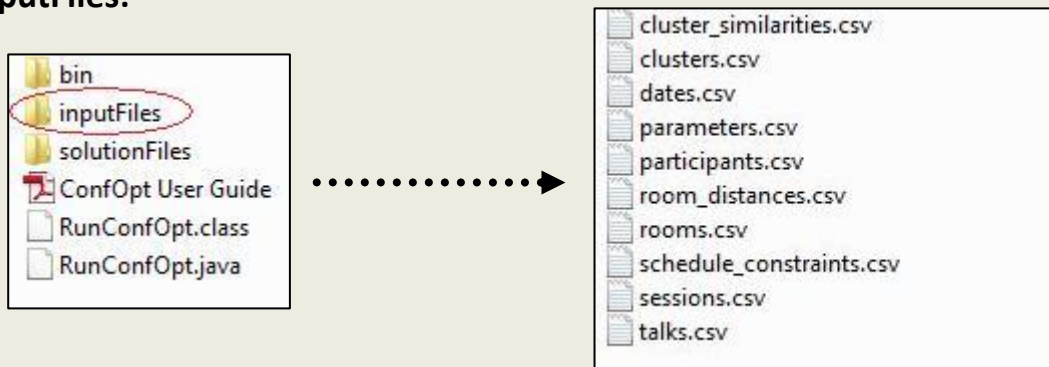
The main input files can be found in the folder “**inputFiles**”. They are written in “**.csv**” format, so you can easily create and edit them in a common *Excel* or *Open Office* table by **semicolon-separation**. Therefore it is necessary that there are no semicolons in your data fields except for separation issues.

Some input files are obligatory to run the program and to some you can add additional data. In the following we give a short description and an example which input format has to be taken for which file.

Notation:

- \mathbb{R} - set of continuous numbers
- \mathbb{N} - set of integer numbers >0

inputFiles:



clusters.csv (required)

Every cluster has to be defined here. The id’s have to be chosen in an interval 1...C, where C is the number of all clusters. So please make sure you have no “gaps” in your id definitions. It is necessary to define at least one cluster.

Format:	id; title;
Data type:	\mathbb{N} ; String;
Example:	15;Optimization in Energy Systems;
Explanation:	Cluster with id 15 is “Optimization in Energy Systems”.

cluster_similarities.csv (required)

This file is used to give each pair of clusters a similarity coefficient, such that “similar” clusters are assigned to nearby rooms. Choose these values between 0 and 1. If this value is 0 then both clusters are regarded as completely different. A value of 1 treats them as equal.

Note:

- For all pairs (even for same clusters, in general = 1) of clusters a similarity value has to be chosen! Otherwise an exception will be thrown.
- Choose them symmetric, because the program will treat them like that.

Format: cluster_id1;cluster_id2;similarity_value;
Data type: $\mathbb{N}; \mathbb{N}; \mathbb{R} \cap [0,1];$
Example: 13;20;0.3;
Explanation: Cluster 13 and cluster 20 have a similarity value of 0.3 .

dates.csv (required)

Each session occurs at certain timeslots. These timeslots are defined here.

Note:

- Make sure you define the id's in the range from 1...T where T is the amount of all timeslots and also choose them consecutively in time. That means time slot t is directly before t+1 ...
- As this program refers to larger conferences you have to choose at least 3 timeslots.

Format: timeslot_id;title;
Data type: $\mathbb{N}; \text{String};$
Example: 3;Monday afternoon;
Explanation: The 3rd timeslot is called “Monday afternoon”.

participants.csv (required)

You need the data of each participant of the conference to define who is talking and who is chairman in which session. It is recommended that each person that is registered for the conference is put in this list.

Note:

- The id's can be chosen arbitrary. Just make sure you do not use them more than once.

Format: id;last name;first name;
Data type: $\mathbb{N}; \text{String}; \text{String};$
Example: 587;Vader;Darth;
Explanation: Participant “Darth Vader” has id 587.

room_distances.csv (required)

When optimizing you need the distances between each pair of rooms. With this data similar streams are preferred to be in nearby rooms.

Note:

- Again for each pair (even for same rooms, in general = 0) a distance has to be chosen. The distances can be chosen arbitrarily.
- Tip: you can put additional penalty values on distances for rooms which are not on the same floor (stair climbing penalty) or which are in different buildings (walking penalty) etc...

Format: room_id1;room_id2;distance_value;
Data type: \mathbb{N} ; \mathbb{N} ; \mathbb{R} ;
Example: 37;12;384.47;
Explanation: The distance between rooms 37 and 12 has a value of 384.47 .

rooms.csv (required)

This is the list of rooms which are available for the conference.

Note:

- Please make sure you have no “id gaps” in the list, same as above.

Format: room_id;room_title;capacity;
Data type: \mathbb{N} ;String; \mathbb{N} ;
Example: 9;H 0110;243;
Explanation: Room “H 0110” has id 9 and a capacity of 243 persons.

schedule_constraints.csv (additional)

In the progress of planning a conference there can occur special schedule requests. They are mostly of some kind of these:

- *interval constraints* – one session is supposed to be scheduled in a certain timeslot interval
- *no overlap constraints* – two sessions must not be scheduled at the same timeslot
- *precedence constraints* – two sessions are supposed to be scheduled in two directly consecutive timeslots and, if possible(see notes), in the same room

All types can be added to this single file.

Note:

- you do not have to set a “no overlap” constraint for persons which are speaker and chairman of different sessions, this is done automatically with the session definitions

- if the capacity is sufficient and the *Precedence*-coefficient (parameters.csv) is not smaller or equal to zero, then the associated precedence-sessions will be scheduled in the same room

Format: **interval:** interval;session;timeslot_min;timeslot_max;
 no overlap: no overlap;session_id1;session_id2;
 precedence: precedence; session_id1;session_id2;

Data type: **interval:** String;ℕ;ℕ;ℕ;
 no overlap: String;ℕ;ℕ;
 precedence: String;ℕ;ℕ;

Example: interval;154;2;7;
 no overlap;47;69;
 precedence;48;47;

Explanation: Session 154 has to be scheduled at the timeslot interval [2,7]. Sessions 47 and 69 must not be scheduled at the same timeslot and in addition session 48 has to be scheduled directly after session 47 and if possible in the same room.

sessions.csv (required)

Each session has to be registered. Also an attractivity value for each session has to be added. These are used to assign more attractive sessions to bigger rooms. The higher the attractivity value, the higher the expected attendance.

Note:

- The id's can be chosen arbitrarily.
- **Do not use 0 as attractivity value**, otherwise there may occur problems in the calculation of expected attendances
- According to your attractivities chosen, the program calculates the expected attendance counts automatically

(!) The expected attendance is proportional to the attractivity values that you have chosen, i.e.:

If you choose one session to be twice as attractive as some other session, the expected attendance will also be twice higher.

You should have an overview of the proportionality for all of your sessions. Heavily over dimensioned attractivity values for some sessions will lead to attendances that will not fit in any room. Choose realistic values.

Format: session_id;session_title;session_cluster_id;session_chair_id;session_value;
Data type: ℕ;String;ℕ;ℕ;ℝ > 0;
Example: 324;Games energy and investment;8;876;3.14159;

Explanation: Session 324 has the title “Games energy and investment”. It belongs to cluster 8 and the chairman is the person with id 876 (*participants.csv*). The session has an attractivity value of 3.14159.

talks.csv (required)

Each session consists of at most 3 talks which have to be registered.

Note:

- The position is the index in which order the talks are given. Make sure you do not use them more than once, otherwise one talk is overwritten.

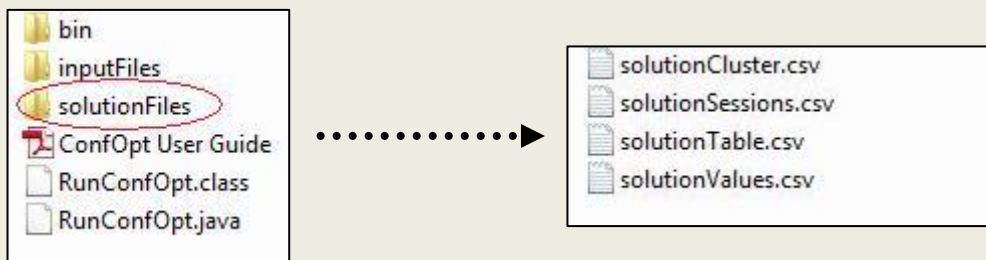
Format: talk_id;talk_title;cluster_id;speaker_id;session_id;position_in_session;

Data type: \mathbb{N} ;String; \mathbb{N} ; \mathbb{N} ; \mathbb{N} ; $\{1,2,3\}$;

Example: 1085; An adaptive layers framework for vehicle routing problems;4;698;23;2;

Explanation: Session 1085 has the title “An adaptive layers framework for vehicle routing problems”. It belongs to cluster 4 and the chair has id 698 (*participants.csv*). This talk will be the 2nd talk in session 23.

5. Output files



The output files can be found in the folder “**solutionFiles**”.

All output-files are generated in a “.txt” format , what you should be able to import in a common table calculation program like **Excel** or **Open Office Calc**. Each file represents a different view of the solution:

solutionCluster.txt

This is the solution regarding the clusters in the room/timeslot matrix. Each number in the table corresponds to the cluster_id you have defined for each cluster.

Extract:

room	capacity	T1	T2	T3	T4	T5	T6	T7	T8	T9	T10	T11	T12	T13	T14	T15
MA 041	99	11	11	11	11	11	11	11	11	11	11	11	11	18	18	18
H 0107	144	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8
MA 005	198	23	23	23	23	23	23	23	23	23	23	23	23	23	23	23
H 0106	99	1	1	1	1	1	1	1	1	1	1	6	6	6	6	6
H 1012	99	5	5	5	5	5	5	5	24	24	24	24	24	24	24	24
H 2053	262	10	10	10	10	10	10	13	13	13	13	13	13	13	13	13
H 3010	231	17	17	17	17	17	17	17	18	18	18	18	18	18	18	18
H 1029	41	14	14	14	14	14	2	2	2	2	2	2	2	2	2	2
H 1058	263	15	15	15	15	15	14	14	14	14	14	14	14	14	14	14

solutionSessions.txt

In this file you see the same matrix replaced by cluster and session title entries.

Extract:

room	capacity	T1
H 3027	80	7 Finance and economics - 241 Financial optimization
MA 042	140	11 Integer and mixed-integer programming - 590 Polyhedral combinatorics
H 2032	235	2 Combinatorial optimization - 354 Constrained clustering
H 1028	235	19 PDE-constrained optimization and multi-level/multi-grid methods - 65 Applications of PDE-constrained optimization
MA 043	152	16 Nonlinear programming - 176 Polynomial optimization and semidefinite programming
H 0110	140	13 Logistics, Traffic and Transportation - 589 Network problems
MA 004	152	9 Global optimization - 632 Advances in global optimization 1
MA 141	70	4 Conic programming - 35 New conic optimization approaches for max-cut and graph equipartition
H 2033	68	2 Combinatorial optimization - 236 Combinatorial optimization in chip design I

solutionValues.txt

The same table with the capacity needed for each session. This table might be useful to see if the sessions fit in the rooms they were assigned to and how the session's attractivity was distributed over the timeslots.

Extract:

room	capacity	T1	T2	T3	T4	T5	T6	T7	T8	T9	T10	T11	T12	T13	T14	T15
MA 041	99	50	69	55	52	64	52	59	55	82	52	27	72	32	43	75
H 0107	144	80	66	78	76	83	72	76	71	47	75	75	65	71	77	81
MA 005	198	80	48	79	64	50	64	55	65	66	52	79	81	65	69	82
H 0106	99	70	80	66	81	57	76	68	70	80	77	65	47	83	60	80
H 1012	99	73	68	63	70	67	73	43	61	75	74	72	83	62	83	76
MA 144	67	62	54	49	52	51	57	53	44	48	44	39	63	42	42	41
H 2038	50	41	20	44	41	45	26	41	45	44	42	23	34	43	47	43
H 3005	80	66	60	61	73	60	64	51	65	62	60	73	70	60	68	74

solutionTable.txt

This is an enlarged table where you can see some details of the sessions, such as id, title and speaker of the talks.

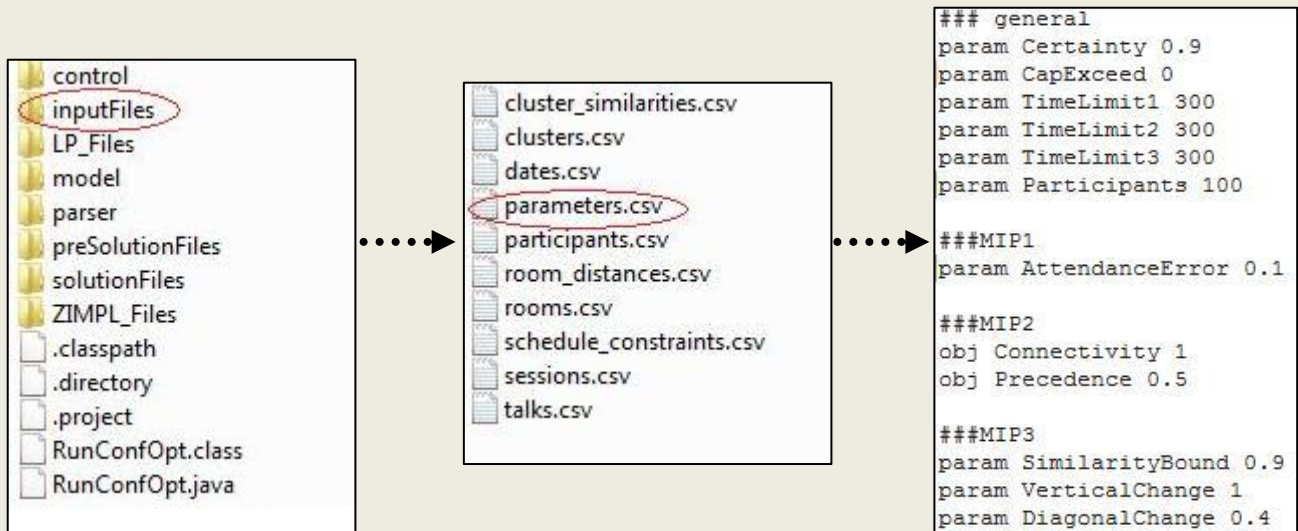
Extract:

room	talk ID	T1
H 3027		7 - Finance and economics : 241 - Financial optimization
talk1	162	Li: A novel method for computing an optimal VaR portfolio
talk2	279	Lin: First-order algorithms for optimal trade execution with dynamic risk measures
talk3	369	Moazeni: Regularized robust optimization for optimal portfolio execution
comment		
MA 042		11 - Integer and mixed-integer programming : 590 - Polyhedral combinatorics
talk1	969	Delle Donne: Vertex coloring polytopes over trees and block graphs
talk2	1054	Forte: Formulations and exact solution algorithms for the minimum two-connected dominating set problem
talk3	1334	Braga: The acyclic coloring polytope
comment		
H 2032		2 - Combinatorial optimization : 354 - Constrained clustering
talk1	485	Borgwardt: On the diameter of partition polytopes and vertex-disjoint cycle cover
talk2	959	Shakhshshneyder: Hardness and non-approximability of constrained clustering
talk3	1646	Brieden: Constrained clustering with convex objective function

6. Further configurations

In the folder “**inputFiles**” you will find a file named “**parameters.txt**”. Here you can add parameters to configure the program. It is differentiated between *parameter*- and *objective* coefficients, which are introduced by a “*param*” or “*obj*” in each line.

a. general parameters



param Certainty:

The model also uses a small stochastic approach. The value stands for the probability that no session will exceed the capacity of the rooms.

Note:

- Higher values will need more capacities.
- A value of 1 is not achievable and will make your MIP's infeasible, because you never have total certainty that capacities will not be exceeded.

Format: param Certainty $\mathbb{R} \cap (0,1)$

Default value: 0.5

Example:

param Certainty 0.99

Explanation:

We want to be sure to 99% that no session will exceed the capacity.

param CapExceed:

You can allow the sessions to exceed the capacities of the rooms to a specific amount. This may also be useful for testing.

Format: param CapExceed \mathbb{N}
Default value: 0

Example:
param CapExceed 5

Explanation:
Each session can take place in a room that has a capacity 5 less than the calculated number of attendees.

param AttendanceError:

Determining the ‘real attendance’ of each session is a nonlinear matter. Due to the linearization of the problem an ‘error’ occurs. You can determine how large this value can be at most.

Note:

- For nearly all realistic cases a value of 0 is not achievable.
- The smaller the error, the better the schedule
- With new settings you should start with a higher (default) value.
- In the end of the computation you will get a comment that shows you on how much this value can still be decreased

Format: param AttendanceError $\mathbb{R} \geq 0$
Default value: 0.1

Example:
param AttendanceError 0.03

Explanation:
The error value is at most 3%.

param TimeLimit1(2,3):

For bigger instances the MIP’s will not being solved optimal in our life cycle.
To avoid high calculation times and for testing purposes you can add time limits to each of the three MIP’s mentioned in section 3.

Format: param TimeLimit1 \mathbb{N}
Default value: 900

Example:
param TimeLimit1 600

Explanation:
The time limit of the “time assignment”-MIP is set to 600 seconds = 10 minutes

param Participants:

Here you can add a value for the number of participants at the conference. This value is independent from the number of participants that you have entered in *participants.csv* .

The number of needed capacities of each session is calculated with this value. This value can also be used to test different degrees of room capacity utilization and worst case scenarios.

Note:

- Too high numbers will end in infeasibility.

Format: param Participants \mathbb{N}
Default value: 100

Example:
param Participants 2013

Explanation:
The conference includes 2013 participants.

b. MIP2 – parameters/objective coefficients (stream forming)

obj Connectivity:

This coefficient is responsible for the time-dependent connection of sessions of the same cluster in the same stream.

Format: obj Connectivity \mathbb{R}
Default value: 1

Example:
obj Connectivity 1

Explanation:
If two sessions of the same cluster are scheduled consecutive in the same stream, the objective will receive a bonus of 1.

obj Precedence:

Increasing this coefficient will prefer sessions with precedence constraints (schedule_constraints.csv) to be scheduled in the same room.

Format: obj Precedence \mathbb{R}
Default value: 0.5

Example:
obj Precedence 3

Explanation:
Having two sessions with a precedence constraint in the same room (and stream), the objective will receive a bonus of 3.

c. MIP3 – parameters/objective coefficients (room assignment)

param SimilarityBound:

This parameter specifies up to which stream-similarity distances are regarded. This coefficient has a very huge influence on the computation time of the 3rd MIP.

Choosing a value of 1 will only look at distances between streams which are cluster-wise completely equal.

A value of 0 will consider all distances between all streams. But here it is recommended to select a value strictly greater than 0 as it will provide better performance. You can iteratively decrease this value when getting fast solutions to obtain better room assignments.

Format: param SimilarityBound $\mathbb{R} \cap (0,1]$
Default value: 0.5

Example:

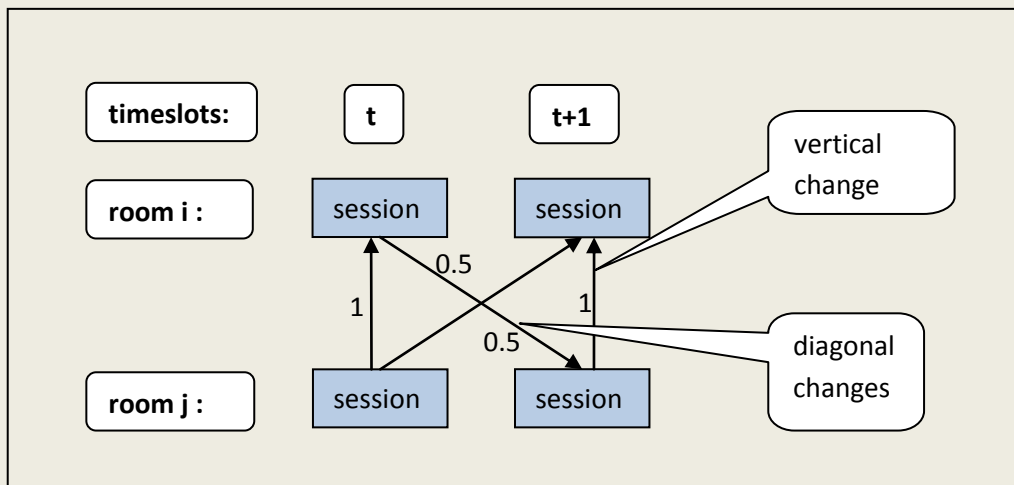
param SimilarityBound 0.5

Explanation:

In the 3rd MIP only considers distances between streams which have a similarity of ≥ 0.5 .

param VerticalChange/DiagonalChange:

These parameters determine the valuation-ratio between room changes. You can differentiate between vertical and diagonal changes:



Format: param VerticalChange \mathbb{R} **Default value:** 1
Format: param DiagonalChange \mathbb{R} **Default value:** 0.5

Example:

param VerticalChange 1
param DiagonalChange 0.5

Explanation:

Room changes at the same time (vertical, changes within a session) are twice as important as room changes between two consecutive sessions.

7. Typical problems

Before reading the following, please make sure that you have taken the right input formats for all input files that were mentioned above. We tried to catch most of the exceptions that can occur and give a short description in the command line, if invalidities are detected.

a. an exception occurs, what am I doing wrong?

- In most cases the input format is violated, check your input data and take a look in the command lines, if any warnings or errors were detected.
- If one of the three MIP's is infeasible, also an exception will be thrown, check b)

b. one of the MIP's is infeasible, what now?

- check, if you have enough room-/timeslots to assign all sessions
- check, if the capacity is sufficient to assign all sessions, try to increase the parameter CapExceed or decrease the number of participants (parameters.csv)
- try to increase the AttendanceError (parameters.csv)
- take a look in your schedule constraints (schedule_constraints.csv) and check, if:
 - one session has to be scheduled into two disjoint intervals
 - you have no 'precedence-cycles'
- check the case, if one person is speaker and chairman, such that his sessions must necessarily overlap at some timeslot
- it is possible that not all schedule constraints can be satisfied
- if MIP 1 produces a feasible solution, MIP 2 and MIP 3 should do as well

c. the computation time and the integrality gaps are very high!

- you can set time limits for each MIP (parameters.csv)
- increase the *SimilarityBound* (for MIP3, parameters.csv)
- from our observations you should get acceptable solutions after:
 - MIP1: <900sec(1-4% gap),
 - MIP2: <600sec (optimal),
 - MIP3: highly depends on the parameter *SimilarityBound* , but when choosing high enough: <600sec(optimal).

d. the schedule looks totally mixed up!

- The capacity constraints can be very restrictive. Increase the parameter CapExceed and try to decrease the AttendanceError to the value obtained from your last solution (shell output line)
- You may take a look at the objective coefficients that you have chosen, the default values (section 6) should work well at first glance, maybe you can try to re-define these values when looking at the solution

- You should also check your cluster similarities, room distances and the attractivity of your sessions, whether they were well-chosen
- Many schedule constraints (schedule_constraints.csv) can be very restrictive to the entire solution
- Set the time limit high enough to find better solutions / integrality gaps

8. Data considered for ISMP 2012

For some data fields, e.g. attractivities, the values can be chosen arbitrary, what gives you a high degree of freedom in your choice of the parameters. We want to present the formulas that we have taken for ISMP 2012 to give you an approach on how a choice of these values can look like.

clusterSimilarities.csv:

As you have to choose similarity values for all pairs of clusters, it can be seen as filling a $C \times C$ -matrix A with entries $a_{ij} \in [0,1]$ where C denotes the amount of clusters. First it is obvious that every cluster has the highest similarity to itself, so we define $a_{ii} = 1 \forall i = 1 \dots C$. Afterwards we did some thematic classification between the clusters by choosing categories, like: Theory, Computation, Application, Discrete, Nonlinear and Stochastic.

For every cluster and every category we choose a value of 1, whether the category corresponds to the cluster and a value of 0 otherwise. Let n_{ij} be the number of categories, which cluster i and j have in common (which are 1 for both) and n_i, n_j the number of categories which apply to i and j (which are equal to 1) we defined $a_{ij} = \lambda \cdot \frac{n_{ij}}{n_i + n_j - n_{ij}} \forall (i,j): i \neq j$ with a factor $\lambda = 0.5$. There are known a lot of other similarity coefficients which might also be useful in this case.

roomDistances.csv

We have to fill a $R \times R$ matrix D with entries d_{ij} between any pair of rooms i and j . For every room we defined coordinates $(x, y, z) \in \mathbb{R}^3$ where x, y correspond to coordinates in the plane according to some coordinate system (e.g. Google Maps, room maps or something similar) and z denotes the floor of the room. Therefore we defined penalty values λ_1 for floor-changes and λ_2 for building-changes. Our building structure was very similar to Manhattan-distances, so finally we chose:

$$d_{ij} = \begin{cases} |x_i - x_j| + |y_i - y_j| + \lambda_1 \cdot |z_i - z_j|, & \text{if } i \text{ and } j \text{ in same building} \\ |x_i - x_j| + |y_i - y_j| + \lambda_1 \cdot |z_i - z_j| + \lambda_2, & \text{otherwise} \end{cases}$$

sessionValues.csv

Finding good values for the attractivity of a session is one of the most challenging parts when planning the schedule. There are many indicators where an attractivity can be recognized. In our considerations we have taken the following values for every session s which belongs to cluster c :

a_s^1 – attendance counts of speakers with high attendance (≥ 40 visitors) at ISMP 2009 (popular speaker bonus)

a_s^2 – number of submitted talks of cluster c

a_s^3 – number of stated preferences in cluster c (entered at webpage registration by each participant)

a_s^4 – average attendance of cluster c at ISMP 2009

$m_s = \frac{|S(c)|}{T}$, where $|S(c)|$ is the number of sessions, which belong to cluster c and T is the number of timeslots

Let $\lambda_1 \dots \lambda_4$ be some weight coefficients.

Attractivity value for session:

$$a_s = \frac{1}{m_s} \cdot (\lambda_1 \cdot a_s^1 + \lambda_2 \cdot a_s^2 + \lambda_3 \cdot a_s^3 + \lambda_4 \cdot a_s^4)$$

9. Lessons learned

Looking back at the time we developed ConfOpt for the ISMP 2012, this program helped a lot to find good schedules according to the data we have considered. But this is exactly the point where some improvement has to be done:

It is very important to have a realistic attractivity-evaluation of each session. From our experience at ISMP 2012 the attractivity mainly depends on the popularity of the speakers in each session. Often this is closely related to the achievements in their field of research, so the topic also may be more interesting. Consequently it is necessary to give each session with popular speakers a proportional higher attractivity value. A possibility to obtain those “popularity-data” is to contact the cluster-chairs, who may know about the popularity of each speaker in their cluster. This will be an essential matter of the complete schedule.

Secondary attractivity indicators may be the number of sessions of a cluster, interest in a specific cluster (entered at webpage registration) and of course the attendance counts of the previous ISMP (or previous conferences of the same type).

If attractivities are underrated, the corresponding sessions will be scheduled in smaller rooms. Finally this will lead to overcrowding what is the most frequent complaint of attending people. Another way to get out of this situation is to choose rooms of size ≥ 50 only. For the average attendance this will be enough to prevent highly overcrowded rooms, when you are out in your attractivity-estimation.

We hope that *ConfOpt* can help you to find solutions that agree with your idea of a good conference-schedule.

For questions, annotations or any bug reports you can refer to:

tesch@zib.de

Have a good scheduling!