GloptLab
a configurable framework for solving continuous, algebraic CSPs

Ferenc Domes, Arnold Neumaier

University of Vienna

2009.09.20 Lisbon, Portugal

Basics

Problem Specification

Algebraic optimization problems

$$\begin{array}{ll} \min & f(x) \\ \text{s.t.} & G(x) \in \mathbf{v}, x \in \mathbf{x}, G(x) \in \mathbf{G}(x) \end{array}$$

with uncertain constraint coefficients can be represented as the quadratic problem

$$\begin{array}{ll} \min & A_{i:}q(\hat{x}) \\ \text{s.t.} & Aq(\hat{x}) \in \mathbf{F} \ \text{ for some } A \in \mathbf{A}, \\ & \hat{x} \in \hat{\mathbf{x}}, \end{array}$$

by introducing intermediate variables. $q(x) := (x, \mathrm{vec}(xx^T))$ is a quadratic monomial vector.

GloptLab is designed to solve such problems, currently for the case when the objective function is constant (CSP).

Introduction

GloptLab

- GloptLab is an **easy-to-use testing and development platform** solving algebraic constraint satisfaction problems, written in Matlab.
- Various new and **state-of-the-art algorithms implemented** in GloptLab are used to reduce the search space.
- All methods in GloptLab are **rigorous**, hence it is guaranteed that no feasible point is lost.
- From the method repertoire **custom made strategies** can be built, with a user-friendly graphical interface.

Methods

Verified Computing

GloptLab uses various **rigorous** methods to bound the feasible domain.

Using the internal form, rigorous means that each method $\Gamma : (\mathbf{x}, \mathbf{F}) \to (\tilde{\mathbf{x}}, \tilde{\mathbf{F}})$ where $\tilde{\mathbf{x}} \subseteq \mathbf{x}$ and $\tilde{\mathbf{F}} \subseteq \mathbf{F}$ has the property

$$\{x \in \mathbf{x} \mid Aq(x) \in \mathbf{F}\} == \{x \in \tilde{\mathbf{x}} \mid Aq(x) \in \tilde{\mathbf{F}}\}.$$

- Rigorous methods reduce the search space while **guarantee that no feasible points are lost**.
- In the applications, serious **safety problems** could **arise from losing feasible points** (Gough platform).

Method Features

- Rigorous methods estimate the error for each step in their algorithms and use **directed rounding** or **interval arithmetic**.
- Another way is to **find approximate solutions and then verify** the results.
- Rigorous computations slow down the solution process, and often require more theoretical work.
- But sometimes having a good approximative solution is not good enough (e.g. computer assisted proofs)!

Method Selection

The following classes of methods are used to rigorously reduce the search space:

- Problem Transformation/Simplification
- Constraint Propagation
- Linear Methods
- Strict Convex Enclosure
- Conic Methods
- Branch and Bound
- Probing, Slicing.

Toolboxes

We make use of several **external toolboxes to compute approximative solutions** of linear, semidefinite or conic programs:

- The toolbox SeDuMi is an optimization tool over symmetric cones developed by Jos F. Sturm.
- Alternatively SDPT3 from Kim-Chuan Toh, Michael J. Todd, and Reha H. Tutuncu.
- Linear programs are solved with LPSolve by Michel Berkelaar.
- Projected BFGS and conjugate gradient methods from C. T. Kelley.

Toolboxes

IntLab, by Siegfried Rump is used for **interval computation** while the AMPL **modeling language** by Robert Fourer, David Gay and Brian Kernighan is used for problem input.

Method References

More details on the implemented methods as well as their mathematical background can be found in various papers on the official GloptLab **homepage**:

http://www.mat.univie.ac.at/~dferi/gloptlab.html

The Features of GloptLab

Summary of the features

- General and **well structured input format**
- Implemented in a **completely modular** way, allowing easy portability of individual methods
- Easy to use for **prototyping** and for **development of new techniques** in the context of other methods
- The **strategy builder** allows us to test different strategies for different problem classes

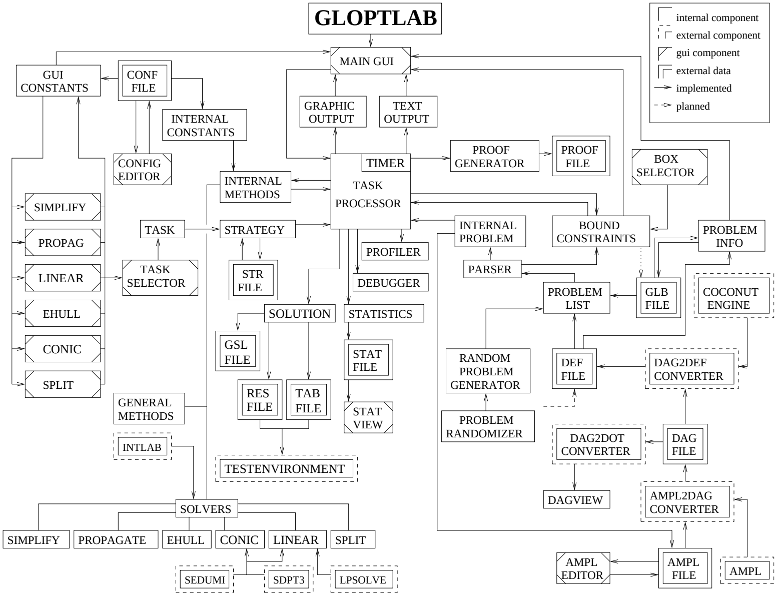
Summary of the features

- **Interactive solution** of a particular problem: stop the execution of the strategy, remove and add new tasks, then resume the solution process
- Contributors can **add new methods with only minimal knowledge** of the other parts of the software
- **Graphical user interface** for **building strategies and visualization** of the solution process
- Batch execution mode, Test Environment compatibility.

GloptLab Structure

GloptLab
a configurable framework for solving continuous, algebraic CSPs

Ferenc Domes, Arnold Neumaier



Strategy building

- In order to solve a problem or a list of problems we need a **strategy**.
- A strategy is a list of **tasks** used to solve a problem.
- A task could be one of the methods listed above, or a **control task**.

Strategy building

- The control tasks like **loops**, **conditions** and **breaks** extend the functionality and **ensure the versatility** of a strategy.
- **Strategies are built comfortably** by using the graphical strategy builder.
- **New methods and solvers are automatically recognized** by the strategy builder.

Simple Sample Strategy

```
Read Problem
Simplify
Feasibility
Begin Condition
   Break
End Condition
Begin While
   Propagate
   Feasibility
   Begin Condition
      Break
   End Condition
End While
Begin Split
   Propagate
   Feasibility
   Begin Condition
      Break
   End Condition
End Split
Merge
Begin Postprocess
   Merge
   Feasibility
End Postprocess
Pause
Finish
```

Complex Sample Strategy

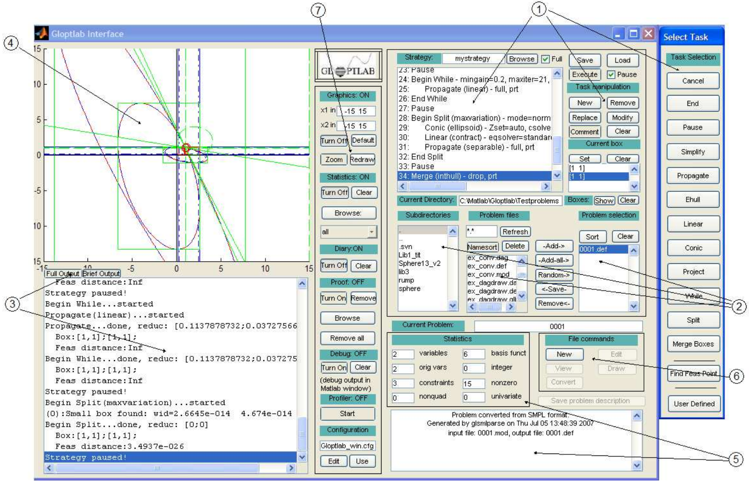

```
Read Problem
Simplify
Ehull
Linear
Feasibility
Begin Condition
   Break
End Condition
Conic
Begin While
   Propagate
   Linear
   Feasibility
   Begin Condition
      Break
   End Condition
End While
Begin Split
   Propagate
   Linear
   Feasibility
   Begin Condition
      Break
   End Condition
End Split
Merge
Begin Split
   Propagate
   Linear
   Feasibility
   Begin Condition
      Break
   End Condition
End Split
Begin Postprocess
   Merge
   Feasibility
End Postprocess
Pause
Finish
```

Problem solving

When a strategy has been built it can be used to solve a specific problem or a list of problems.

Solving can be started either by using the batch solution mode or directly in the **graphical user interface** of GloptLab:

Graphical user interface



Graphical user interface

The graphical user interface consists of **areas for entering problems**, for **defining strategies**, for **displaying the solver progress** and for configuring GloptLab.

The interactive solution of a particular problem in the graphical user interface: it is possible to **stop the execution** of the strategy, remove and **add new tasks** to it and then **resume** the solution process.

Manipulating the method parameters, **experimenting with different combinations of tasks** can greatly improve the solution results and lead to more knowledge about **building effective strategies**.

Demonstration

Test Conditions

We used the TEST ENVIRONMENT to test and compare some GLOPTLAB strategies.

Library LIB3 of the COCONUT Environment Testset containing **308 constraint satisfaction** problems has used, **63 of them was classified as hard** problems the other as easy ones.

The two sample strategies have been configured to accept only problems with less than 100 variables and used to solve the library.

The **maximal time allowed** for the solution of a single problem was **120 seconds**.

Test Results of the Simple Sample Strategy

GloptLab on Lib3 (Simple Sample Strategy)								
stat	all	wr	easy loc.			hard loc.		
			+G	-G	I	+G	-G	I
all	308	0	121	124	0	14	29	20
G	125	0	111	0	0	14	0	0
X	76	0	0	57	0	0	8	11
TU	95	0	8	59	0	0	21	7
U	12	0	2	8	0	0	0	2

GloptLab summary statistics						
lib	all	accept	+G	G!	G?	I?
Lib3	308	232	135	125	0	0

Test Results of the Complex Sample Strategy

GloptLab on `Lib3` (Complex Sample Strategy)								
stat	all	wr	easy loc.			hard loc.		
			+G	-G	I	+G	-G	I
all	308	0	130	115	0	19	24	20
G	139	0	120	0	0	19	0	0
X	76	0	0	57	0	0	8	11
TU	85	0	10	52	0	0	16	7
U	8	0	0	6	0	0	0	2

GloptLab summary statistics						
lib	all	accept	+G	G!	G?	I?
Lib3	308	232	149	139	0	0

Evaluation of the Test Results

135 correct solutions found (125 of them was claimed as correct) by using the **first strategy**

149 correct solutions (139 of them was claimed as correct) by using the second **second strategy**

Within the same allowed solution time we solved approximately 10 percent more problems with the second strategy than with the first one.

35 percent more hard problems was solved by using the second strategy!

The significant difference was caused by the more sophisticated methods and the clever structure of the second strategy.

Perspectives and Conclusions

Perspectives

- Integrating the non algebraic, univariate functions.
- Testing and improving the optimization part.
- Enhancing the existing methods and developing new ones.
- Implementing promising methods in the COCONUT Environment.
- Comparison with other solvers (ICOS, Realpaver, Baron, GlobSol, etc.)
- Automatic, intelligent strategy selection.

Conclusions

External contributors are welcome to join the project by implementing and testing their own user-defined methods. User-defined methods submitted to us will be permanently added to the method repertoire of future versions of GloptLab if they are promising enough.

I would like to thank Arnold Neumaier for his help and support.

This research was founded by the grant FSP 506/003 of the University of Vienna.

Conclusions

Thank You for your attention!

If you have question about GLOPTLAB please contact me during the CP2009 Conference or send me an e-mail to:
`ferenc.domes@univie.ac.at`

You are welcome to **test and play** with the current version of GLOPTLAB, downloadable from:
`http://www.mat.univie.ac.at/~dferi/gloptlab.html`