Creation of Conflict Graphs for Integer Programming: a Computational Study

Samuel Souza Brito [1]

Computing Department
Universidade Federal de Ouro Preto - UFOP
Ouro Preto, Brazil

Haroldo Gambini Santos [2]

Computing Department
Universidade Federal de Ouro Preto - UFOP
Ouro Preto, Brazil

Abstract

This is a short example to show the basics of using the ENDM style macro files. Ample examples of how files should look may be found among the published volumes of the series at the ENDM home page (`http://www.elsevier.com/locate/endm`)

Keywords: Please list keywords for your paper here, separated by commas.

1 Introduction

In this work we present an approach for the creation of conflict graphs for Integer Programming (IP) problems. A conflict graph represents logical re-

[1] Email: `samuelsouza@iceb.ufop.br`
[2] Email: `haroldo@iceb.ufop.br`

lations between binary variables. This kind of graph has a vertex for each binary variable and its complement. An edge between two vertices indicates that the variables involved, represented by the vertices, can not be activated (assigned with value 1) at the same time without violating some constraint of the problem.

Conflict graphs are typically constructed using probing techniques [3] based on constraints analysis. The probing technique consists in analyzing logical implications generated by fixing binary variables. For example, in a given problem the activation of the variable x implies the deactivation of the variable y, in order to respect the constraints of the problem. In this case, we found a logical relation of the form "$x = 1 \rightarrow y = 0$". This logical relation can be represented in a conflict graph, creating an edge between the vertice that represents the variable x and the vertice that represents the complement of the variable y.

Building a graph by looking for pairwise of conflicts may be computationally prohibitive when the input problem is large. Thus, the computational efficiency of this technique depends on the complexity of the constraint exploration, causing a trade off between efficiency and effectivity. We also use probing techniques based on feasibility considerations, by analysing constraint-by-constraint. However, we compute the bounds of the left-hand side for each constraint and compare with the respective right-hand side, in order to avoid the analysis of constraints that would not lead to the discovery of conflicts. Furthermore, this comparison allows the detection of constraints that form cliques, avoiding the pairwise analysis of the variables in these constraints. The use of such strategy contributes to accelerate the conflict discovery process as shown in the performed experiments.

An important application for the use of conflict graphs is the clique cut generation, which performs a crucial role in the discovery of strong inequalities [4] in IP problems. The dynamic inclusion of these inequalities allows tightening the linear relaxation of an IP problem, crucial to the branch-and-bound based solvers [2]. Moreover, conflict graph can also be used to develop heuristics and branching primal schemes. Hoffman and Padberg [6] used conflict graphs to generate valid inequalities for set partitioning problems arising in airline crew-scheduling. Achterberg [1] presented heuristics based on SAT techniques for Mixed Integer Programming solvers to generate valid inequalities from the current infeasible subproblem and the associated branching information. The same idea has been developed independently and in parallel by Sandholm and Shields [8]. With our experiments we can demonstrate that the generation and insertion of clique cuts contribute significantly to strengthening the linear re-

laxation, especially on problems that have constraints on the type Generalized Upper Bound (GUB constraints).

For ease of understanding of our approach we consider only pure binary IP problems. Despite this, it can be applied to any IP problem containing binary variables. The rest of the paper is organized as follows. In Section 2, we formally explain our approach to build conflict graphs as well our strategy to speed the detection of logical implications. In Section 3, we present the clique cut separation routine, including a clique extension step. In Section 4, the computational experiments with MIPLIB 2010 instances [7] and instances of the International Nurse Rostering Competition [5] are presented and analyzed. Finally, in Section 5, we conclude and give future directions about this work.

2 Conflict Graphs in Integer Programming

3 Clique Cut Separation

4 Experimental Results

5 Conclusions and Future Work

6 Frontmatter

The biggest difference between a "usual" LaTeX style such as `article.sty` and the ENDM package is that the ENDM macro package requires the title, author's name or names, abstract, keywords and "thanks" all to be included within the `frontmatter` environment. At the beginning of the source file for this paper, you'll notice this. Also, you'll notice that the usual `\maketitle` is absent; it no longer is needed. The ENDM style package automatically generates the title, author's name and address, and related material at the beginning of the paper. Note also that hyperref has been disabled in this part of the endm.cls file, so references to footnotes aren't linked to the appropriate footnotes or addresses. This is an old problem with LaTeX, involving the fact that the references within the frontmatter aren't passed cleanly to the linking software.

For those who have used the ENDM package before, the one new thing to note is the inclusion of *Keywords* which are now required by Elsevier.

The ENDM macro package provides two alternatives to listing authors names and addresses. These are described in detail in the file `instraut.pdf`. Basically, listing each author and his or her address in turn, is the simplest

method. But, if there are several authors and two or more share the same address (but not all authors are at this address), then the method of listing authors first, and then the addresses, and of referencing addresses to authors should be used.

Furthermore, note that an acknowledgment of support (the contents of `\thanks`) should be done by a separate listing of `\thanks[NSF]{To the NSF}` with the optional argument – `[NSF]` – being used for `\thanksref` which is attached to those authors acknowledging such support. It is important that the `\thanks` not be included within the scope of `\author{}` or of `\title{}`, but it must be within the scope of the environment `frontmatter`.

More details about added terms such as `\collab` can be found in the file `instraut.pdf` if they are needed.

7 Sectioning and Environments

Since ENDM is published through the auspices of Elsevier B. V., their style files were used to create the ENDM macro package. Below is a proof which shows that this package is not much different to most others:

Definition 7.1 A file is *derived* from another file if it was obtained by making only a few modifications to the original file.

Theorem 7.2 *The file* `endm.cls` *is derived from* `elsart.sty`.

Proof. This is clear from the similarity of the output to the output from the standard Elsevier style files. □

If one wants to start a proof with a descriptive word, such as "sketch", then one can use the `\begin{proof*}...\end{proof*}` environment, as in

Proof (Sketch) This can be derived from simple observations. □

The main difference between the file `endm.cls` and the `elsart.cls` file used for other Elsevier journals is the more precise format we use. Elsevier's generic style files are meant for preliminary editing and more precise formatting is imposed using a macro file designed for the specific Elsevier journal in which the paper will eventually appear. The `endm.cls` and `endmmacro.sty` files format papers uniformly so that they all are easily recognizable as belonging to the series *Electronic Notes in Discrete Mathematics*.

All of the usual features of LaTeX are available with these style files. It is only the formatting that has been rigorously defined. One can use the sectioning commands `\section,\subsection, \paragraph` and `\subparagraph`.

The numbering scheme used is one under which Theorem 1.2.3 is the third numbered item in the second subsection of the first section of the paper. In order to facilitate cross-references, all of the named environments given below are numbered and all use the same numbering scheme.

The file `endmmacro.sty` contains additional information that is needed to typeset a paper. It also has the definitions of the $\mathcal{AMS}$ `euler` and `blackboard bold` fonts builtin. If you want to use symbols for the natural numbers, the reals, etc., then we prefer that you use the blackboard bold fonts, and not plain bold fonts. This is accomplished by using the `\mathbb` font, as in $\mathbb{N}$ or $\mathbb{R}$.

The names of theorem-like environments are provided in `endmmacro.sty`. With the exception of the environment "Algorithm", the names of all these are the full name rather than a shortened version. The environments provided and their names are as follows:

- `\begin{theorem} ... \end{theorem}` for Theorems,
- `\begin{lemma} ... \end{lemma}` for Lemmas,
- `\begin{corollary} ... \end{corollary}` for Corollaries,
- `\begin{proposition} ... \end{proposition}` for Propositions,
- `\begin{criterion} ... \end{criterion}` for Criteria,
- `\begin{alg} ... \end{alg}` for Algorithms,
- `\begin{definition} ... \end{definition}` for Definitions,
- `\begin{conjecture} ... \end{conjecture}` for Conjectures,
- `\begin{example} ... \end{example}` for Examples,
- `\begin{problem} ... \end{problem}` for Problems,
- `\begin{remark} ... \end{remark}` for Remarks,
- `\begin{note} ... \end{note}` for Notes,
- `\begin{claim} ... \end{claim}` for Claims,
- `\begin{summary} ... \end{summary}` for Summary,
- `\begin{case} ... \end{case}` for Cases, and
- `\begin{ack} ... \end{ack}` for Acknowledgements.

For example,

Algorithm 1 *Step 1: Write the paper*
Step 2: Format it with the ENDM macro package
Step 3: Ship the whole thing to the Guest Editors

8 References and Cross-references

All the cross-referencing facilities of LaTeX are supported, so one can use `\ref{}` and `\cite{}` for cross-references within the paper and for references to bibliographic items. As is done in this note, the *References* section can be composed with `\begin{thebibliography}...\end{thebibliography}`. Alternatively, BibTeX can be used to compile the bibliography. Whichever one is used, the references are to be numbered consecutively, rather than by author-defined acronyms. Of course you can use your own acronyms for easy reference to each of the items in the bibliography, as has been done with the listing for this short note.

Note that the references should *not* be started with a new `\section` command.

The package `hyperref` is automatically loaded by endm.cls and this makes all the cross-references within the document "active" when the pdf file of the paper is viewed with Adobe's Acrobat© Reader. The format for including a link is simple: simply insert `\href{URL} {text}` where *URL* is the URL to which you want the link to point, and *text* is the text you want to be highlighted and which will bring up the desired web page when clicked upon.

8.1 Particulars about `.pdf` files

We now require that `.pdf` files be provided for publication online. A `.pdf` file is viewable by Adobe's Acrobat© Reader, which can be configured to load automatically within a browser. Viewing a properly formatted `.pdf` file with Acrobat© allows the cross-references and links to URLs to be active. In fact, Elsevier utilizes `.pdf` files in order to take better advantage of the web's capabilities.

One point that needs to be emphasized is that you should use Type 1 fonts when you typeset your LaTeX source file. These fonts are scalable, meaning that they carry information that allows the device viewing the final output to scale the fonts to suit the viewer being used (from an onscreen viewer such as Adobe's Acrobat© Reader to printing the file on a printer). You can tell if you have used the right fonts by viewing the final output on your machine. If the font looks grainy, then you have not used a Type 1 font. Type 1 fonts can

be located at the CTAN archive at http://www.ctan.org. They are public domain fonts and do not cost anything when you add them to your system.

Assuming you have Type 1 fonts available, there are several methods for producing .pdf files.

Using dvips and ps2pdf

We list this option first since it appears to be the most reliable and the easiest to use, especially if you include embedded PostScript graphics (.eps files) in your source file. Simply run LaTeX2e on your source file, apply dvips to produce a PostScript file and then apply ps2pdf to obtain a .pdf file.

The DVIPDFM utility

Another easy method for producing acceptable .pdf files is via the utility dvipdfm. This utility is included in distributions of MikTeX, which runs on Windows machines, but it probably needs to be added to your teTeX distribution, if you are running LaTeX on a UNIX machine. The utility and precise information about installing it on your system can be found at the web page http://gaspra.kettering.edu/dvipdfm/. In essence, this utility converts a .dvi file into a .pdf file. So, one can first prepare the .dvi file using LaTeX, and then apply the utility dvipdfm to produce the needed .pdf file.[3] This utility makes the inclusion of graphics particularly simple. Those that are included in the LaTeX source file are simply converted to the .pdf format. As we note below, things are not so simple with the second alternative, which is to use pdfLaTeX.

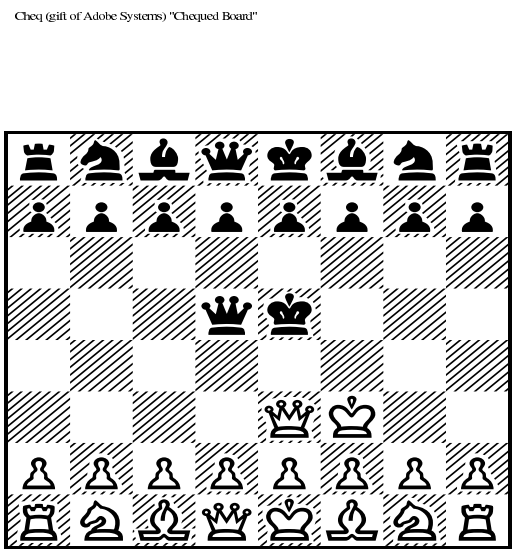
pdfLaTeX

An alternative to the first possibilities to produce .pdf files is to process the source file with pdfLaTeX. This format is available from the standard CTAN sites http://www.ctan.org. It appears that pdfLaTeX and hyperref have some problems when used together. It is necessary to use pdfLaTeX version 14d or later in order to minimize these issues. If your system has an earlier version (most teTeX distributions have version 13d), then you can update your system by retrieving the latest version of pdfLaTeX from ftp://ftp.cstug.cz/pub/tex/local/cstug/thanh/pdftex/. Even if the recent versions are used, pdfLaTeX has the same dealing with references embedded with the frontmatter section described above for LaTeX.

[3] *Beware*! The utility dvipdf does *not* produce acceptable .pdf files, and should not be used. Only dvipdfm should be used to produce .pdf files.

But there is one aspect of pdfLaTeX that creates problems. Many authors include EPS[4] files within their papers. While this is fairly straightforward with LaTeX, there are a couple of points to note when attempting this with pdfLaTeX.

To include a PostScript image in a `.pdf` file produced with pdfLaTeX, you first have to convert the image to a `.pdf` file. The conversion can be accomplished most easily using Ghostscript; you can simply view the file in Ghostview and then print the image to a `.pdf` file using the `pdfwriter` option within Ghostview. The result for a standard chess board that is part of the Ghostview distribution is the following image:



Below is a copy of a color image. While pdfLaTeX can handle image files in other formats, LaTeX can only handle `.eps` images reliably.

[4] EPS stands for *embedded PostScript*, which affords a mechanism for including pre-prepared PostScript files within a LaTeX document.

Using ENDM Macros with Mac OS X

Clearly, if your file does not require `.eps` or other PostScript files, then you can create the required `.pdf` file using any of the standard TEX implementations for the Macintosh. If you do need to include PostScript files and if you are using TEXShop, then you can specify to use dvips and Ghostview in processing your file, and then you can apply `ps2pdf` to create the needed `.pdf` file. Alternatively, the Mac OS X operating system is based on UNIX, so it supports the use of teTEX as described above.

9 Summary and Remarks

The ENDM macro package is relatively easy to use and provides a uniform layout for all the papers that appear in ENDM.

Assigning Volume Numbers

An additional point worth mentioning is that ENDM has moved to *ScienceDirect*, Elsevier's main platform for publishing electronic series. Because *ScienceDirect* cannot easily accommodate changes to published material, the *Proceedings* must be entirely ready before they can be published. Volume

numbers will therefore not be assigned for the *Proceedings* until the final versions of all papers are in.

Copyright Transfer Forms

Due to the move to *ScienceDirect*, the corresponding author of each paper published in ENDM must submit a signed Copyright Transfer Form to Elsevier in order for their paper to be published. A copy of this form will be sent to each author. Note that the publication of an abstract or extended abstract in ENDM will not restrict the author(s) from publishing a full-length article on the same topic and with the same title in another journal (possibly with another publisher). Details about the copyright agreement specifying the exact rights of the authors and the rights of Elsevier are available at Elsevier's Author Gateway.

10 Bibliographical references

ENDM employs the `plain` style of bibliographic references in which references are numbered sequentially and listed in alphabetical order according to the first author's last name. Please utilize this style. We have a BibTEX style file, for those who wish to use it. It is the file `endm.bst` which is included in this package. The basic rules we have employed are the following:

- Authors' names should be listed in alphabetical order, with the first author's last name listed first followed by initials or first name, and with the other authors' names listed as *first name, last name*.
- Titles of articles in journals should be in *emphasized* font.
- Titles of books, monographs, etc. should be in quotations.
- Journal names should be in plain roman type.
- Journal volume numbers should be in boldface, immediately followed by the year of publication enclosed in parentheses in roman type.
- References to URLs on the net should be "active" and the URL itself should be in `typewriter` font.
- Articles should include page numbers.

The criteria are illustrated by the examples below.

References

[1] Achterberg, T., *Conflict analysis in mixed integer programming*, Discrete Optimization **4** (2007), pp. 4 – 20, mixed Integer Programming {IMA} Special Workshop on Mixed-Integer Programming.

[2] Atamtrk, A., G. L. Nemhauser and M. W. Savelsbergh, *Conflict graphs in solving integer programming problems*, European Journal of Operational Research **121** (2000), pp. 40 – 55.

[3] Borndorfer, R., "Aspects of set packing, partitioning, and covering," Ph.D. thesis (1998).

[4] Chvátal, V., *Edmonds polytopes and a hierarchy of combinatorial problems*, Discrete mathematics **4** (1973), pp. 305–337.

[5] Haspeslagh, S., P. de Causmaecker, A. Schaerf and M. Stølevik, *The first international nurse rostering competition 2010*, Annals of Operations Research (2012), pp. 1–16.

[6] Hoffman, K. L. and M. Padberg, *Improving lp-representations of zero-one linear programs for branch-and-cut*, ORSA Journal on Computing **3** (1991), pp. 121–134.

[7] Koch, T., T. Achterberg, E. Andersen, O. Bastert, T. Berthold, R. E. Bixby, E. Danna, G. Gamrath, A. M. Gleixner, S. Heinz, A. Lodi, H. Mittelmann, T. Ralphs, D. Salvagnin, D. E. Steffy and K. Wolter, *MIPLIB 2010*, Mathematical Programming Computation **3** (2011), pp. 103–163.

[8] Sandholm, T. and R. Shields, *Nogood learning for mixed integer programming*, in: *Workshop on Hybrid Methods and Branching Rules in Combinatorial Optimization, Montréal*, 2006.