Anwendung der linearen Optimierung

Power-LP 0.7.4

Robin Götz 288119

Julian Vögt 288840

Inhaltsverzeichnis

[Abbildungsverzeichnis 2](#_Toc440891392)

[1. Einleitung 3](#_Toc440891393)

[2. Projektbeschreibung 4](#_Toc440891394)

[3. Umsetzung 5](#_Toc440891395)

[3.1 Eingesetzte Software 5](#_Toc440891396)

[3.2 Entfernen der Registry 5](#_Toc440891397)

[3.3 Erstellung des Power-LP Free 7](#_Toc440891398)

[3.4 Anpassung der Versionsnummer 9](#_Toc440891399)

[4. Zusatzaufgaben 11](#_Toc440891400)

[4.1 Einbinden des GLPK-Solvers 11](#_Toc440891401)

[4.2 Korrektes Ausführen von Power-LP-Dateien 12](#_Toc440891402)

[5. Probleme und Einschränkungen der Software 14](#_Toc440891403)

# Abbildungsverzeichnis

[Abbildung 1 - Eingabefenster Power-LP 7](#_Toc440891383)

[Abbildung 2 - Lösungsansicht des Power-LP Free 8](#_Toc440891384)

[Abbildung 3 - Neue Titelleiste 9](#_Toc440891385)

[Abbildung 4 - Neue Infobox des Power-LP 9](#_Toc440891386)

[Abbildung 5 - Änderung der Infobox 10](#_Toc440891387)

# Einleitung

Im Rahmen der Veranstaltung „Anwendung der linearen Optimierung“ bei Herrn Prof. Dr. Grütz sollen verschiedene Methoden und Solver der Methodenbank angepasst und optimiert werden. Jeder Studierende hat die Aufgabe drei Methoden zu analysieren und die Ergebnisse zu präsentieren. Anschließend sollen in Zweierteams Methoden und Solver weiterentwickelt werden. Das Projekt ist in einer Dokumentation festzuhalten und schließt mit einer Präsentation ab.

# Projektbeschreibung

Aufgabenstellung

Bei jeder Installation von Power-LP müssen sämtliche Pfade neu gesetzt werden. Dies führt zu Problemen, weil die Pfade dieses Solvers in der Registry des Computers abgelegt werden. Die Methode soll unabhängig lauffähig sein. Die Ausführung des Power-LP soll unabhängig der Registry und ohne das manuelle Ändern von Pfaden in den Einstellungen des Power-LP möglich sein.

Des Weiteren wird Power-LP im OR Web angeboten. Seit Ende letztem Semester gibt es eine neue Version des OR Web. Hierbei haben hochschulinterne Zugang zu allen Methoden und Solver. Externe jedoch können nur lizenzrechtlich freie Software nutzen. Aus diesem Grund sollen in diesem Projekt zwei Versionen des Power-LP entstehen. Eine Power-LP Free Version, die nur lizenzfreie Solver enthält und eine Power-LP Version mit lizenzpflichtigen Solvern.

# Umsetzung

## Eingesetzte Software

Zur Bearbeitung des Power-LP wurde die Software Borland C++ Builder 6 verwendet. Dabei handelt es sich um eine Entwicklungsumgebung, mit der auch grafische Oberflächen bearbeitet werden können wie sie bei Power-LP verwendet werden. Der C++ Builder ermöglicht auch das Kompilieren der Dateien, sowie das sofortige Ausführen des erstellen Programms. Da Power-LP auf grafische Elemente der C++ Builder-Bibliothek zurückgreift, musste die Entwicklung zwingend mit dieser Software erfolgen. Bei anderen Entwicklungsumgebungen wie z.B. Eclipse traten Fehlermeldungen auf.

## Entfernen der Registry

Zur Entfernung der Registry und zum automatischen Anpassen der Solver-Pfade wurde versucht den folgenden Ansatz umzusetzen: Beim Starten von Power-LP soll das Programm herausfinden in welchem Verzeichnis die Anwendung ausgeführt wird. Anschließend sollen die Pfade zu den Solvern und zu anderen Ordnern angepasst werden.

In der Datei registry.cpp wurden zunächst die Funktionen angepasst, die den Solver-Pfad ausgeben. Der Pfad war vorher im Programmcode festgelegt. Nun sollte er variabel gemacht werden.

Der nachfolgende Code zeigt beispielhaft die Ausgabe des Pfades zum XA-Solver

char\* \_\_fastcall reg\_getxadir() {

char result[ MAX\_PATH ];

std::string str = (std::string( result, GetModuleFileName( NULL, result, MAX\_PATH ))).c\_str();

str.resize(str.size()-11);

str += "Solver\\XA\\";

char tab2[1024];

strncpy(tab2, str.c\_str(), sizeof(tab2));

tab2[sizeof(tab2) - 1] = 0;

return tab2;

}

Es wird zunächst der Pfad gesucht in welcher die powerlp.exe ausgeführt wird. Dieser Pfad wird als String gespeichert. Nun werden die letzten 11 Zeichen des Pfades, also "powerlp.exe" entfernt. Anschließend wird "\Solver\XA" angehängt. Der String wird danach in ein Character Array umgewandelt, da die anderen Funktionen, welche reg\_getxadir() aufrufen diesen Datentyp zwingend benötigen. Die Ausgabe des Pfades und deren Verwendung bereitete große Schwierigkeiten. Es wurde anfangs keine Möglichkeit gefunden Zeichen an ein Character Array anzuhängen. Daher wurde ein String gewählt. Nun gab es aber Probleme mit dem Datentyp String und die Umwandlung in ein Character Array. Es mussten viele verschiedene Codevarianten ausprobiert werden und es war sehr viel Recherchearbeit notwendig.

Bei der Ausführung von Power-LP bzw. beim Lösen eines LP-Ansatzes wird die Funktion reg\_getworkdir() oft aufgerufen. Dies verursachte Fehlermeldungen, die zunächst nicht erklärt werden konnten. Es wird vermutet, dass das schnelle, mehrfache Aufrufen der Funktion mehrmals hintereinander diesen Fehler verursacht, da der Pfad zu lang wird und somit nicht mehr richtig gespeichert werden kann.

Dieses Problem wurde umgangen in dem beim Programmstart nun einmalig die Funktion reg\_setworkdir() aufgerufen wird. Diese holt sich den Pfad des Temp-Ordners und speichert ihn in einer globalen Hilfsvariable ab. Die Funktion reg\_getworkdir() holt nun nur noch den bereits gespeicherten Pfad und gibt nur diesen zurück. Eine ständige Überprüfung in welchem Verzeichnis powerlp.exe ausgeführt wird fällt daher weg.

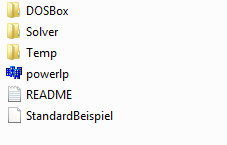
Alle anderen Funktionen, welche die Solver-Pfade in die Registry speicherten wurden entfernt.

In der Datei solver.cpp mussten diverse hart codierte Pfade geändert werden und diese variabel gemacht werden. Zudem war es nötig die von den Solvern erstellten Dateien wie z.B. Batchdateien zu überprüfen und eventuell die Pfade daran angegebenen Pfade anzupassen.

Es musste sehr viel Zeit in Fehlersuche investiert werden, da nach der Durchführung der beschriebenen Änderungen die Solver nicht funktionierten. Die Lösung des Problems war folgende: Der Ordner, in dem sich der Source-Code befand, hieß während der Entwicklungsarbeit "Power LP". Das darin enthaltene Leerzeichen verursachte vermutlich beim Auslesen des Pfades Probleme. Nach der Entfernung des Leerzeichens funktionierten alle Solver bis auf den Weidenauer-Optimizer. Wieder war eine lange Fehlersuche notwendig. Dabei kam heraus, dass der Weidenauer-Solver nicht funktioniert, wenn dessen Anwendung sich in einem zu langen Dateipfad befindet. In Abschnitt 5 der Dokumentation wird genauer darauf eingegangen.

Um den Pfad für den Weidenauer-Solver zu verkürzen, wurde der Ordner in dem sich der Solver befindet von "Weidenau Optimizer" in "Weidenauer" umbenannt. Zudem wurde eine Readme-Datei geschrieben, die den Benutzer auf dieses Problem hinweist.

Die Ordnerstruktur innerhalb des Ordners „PowerLP“, wie sie auf Abbildung X zu sehen ist muss zwingend beibehalten werden.



Die hier zu sehenden Ordner dürfen weder verschoben noch umbenannt werden. Dies betrifft ebenfalls alle Unterordner. Grund ist folgender: Es ist möglich den Pfad bis zu dem Ordner von Powerlp.exe variabel zu halten. Der Pfad zu den verschiedenen Solvern muss ab diesem Ordner aber hart codiert werden. Ansonsten werden die Solver nicht gefunden. Auch dieser Umstand ist in der Readme-Datei vermerkt

## Erstellung des Power-LP Free

Da bei der lizenzfreien Version von Power-LP nicht alle Solver verfügbar sein dürfen, mussten diese entfernt werden. Abbildung 1 zeigt die Auswahlboxen für die verschiedenen Solver mit denen eine Berechnung durchgeführt werden kann. Auf der linken Seite ist zu sehen wie es bei Power LP 0.7.3 ist. Die rechte Seite zeigt Power-LP Free. Bei Power-LP Free wurden die Auswahlboxen der lizenzpflichtigen Solver entfernt.

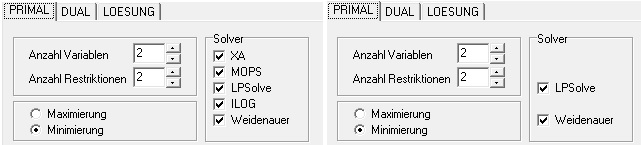


Abbildung 1 - Eingabefenster Power-LP

Die Lösungsansicht wurde bei Power-LP Free ebenfalls angepasst. Wie auf Abbildung 2 zu sehen ist, werden nun nur noch die Lösungen der lizenzfreien Solver angezeigt. Ein Löschen der einzelnen Reiter ist leider nicht möglich. Daher musste das Fenster neu erstellt werden.

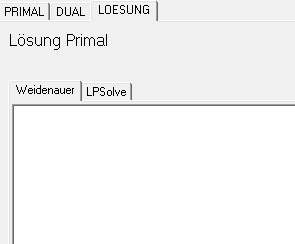


Abbildung 2 - Lösungsansicht des Power-LP Free

Der unten aufgeführte Quellcode aus der Datei ChildWin.cpp ist dafür zuständig, die Solver auszuführen, deren Auswahlbox aktiv ist. Da die Auswahlboxen der Solver XA, ILOG und MOPS entfernt wurden, wurde auch dieser Teil des Quellcodes entfernt (rot markiert).

if (CheckBoxXA->Checked) {

MemoXA->Lines->Clear();

Solver->RunXA(CLP);

if (!Solver->getXAFile(filename, sizeof(filename))) {

LoadFromFile(MemoXA, filename);

}

}

if (CheckBoxMOPS->Checked) {

Solver->RunMOPS(CLP);

if (!Solver->getMOPSFile(filename, sizeof(filename))) {

LoadFromFile(MemoMOPS, filename);

}

}

if (CheckBoxLPSolve->Checked) {

Solver->RunLPSolve(CLP);

if (!Solver->getLPSolveFile(filename, sizeof(filename))) {

LoadFromFile(MemoLPSolve, filename);

}

}

if (CheckBoxILOG->Checked) {

Solver->RunILOG(CLP);

if (!Solver->getILOGFile(filename, sizeof(filename))) {

LoadFromFile(MemoILOG, filename);

}

}

if (CheckBoxWeidenauer->Checked) {

Solver->RunWeidenauer(CLP);

if (!Solver->getWeidenauerFile(filename, sizeof(filename))) {

LoadFromFile(MemoWeidenauer, filename);

}

}

Bei Power-LP Free wurde außerdem die Titelleiste des Programms geändert.

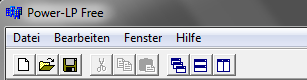


Abbildung 3 - Neue Titelleiste

Zuletzt wurde die HTML-Seite, mit der Beschreibung des Power-LP für OR-Alpha, angepasst. Hinweise für nicht mehr verwendete Solver wurden entfernt und es wurde ein neuer Screenshot des Power-LP Free hinzugefügt.

## Anpassung der Versionsnummer

Da durch das Projekt eine neue Version des Power-LP entstand, musste die Versionsnummer aktualisiert werden. Abbildung 4 zeigt die neue Infobox des Power-LP mit der Versionsnummer 0.7.4. Es wurde zudem hinzugefügt, von wem die neue Version erstellt und mit welcher Software dies umgesetzt wurde.

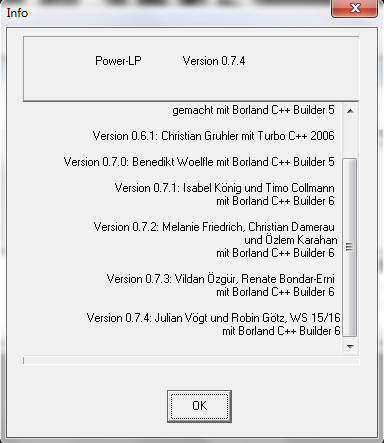


Abbildung 4 - Neue Infobox des Power-LP

Abbildung 5 zeigt wie die Änderung der Infobox im C++ Builder durchgeführt wird. Nach anwählen der "AboutBox" im Project Manager öffnet sich der Object Inspector, wie er auf der linken Seite zu sehen ist. Klickt man nun auf den Button neben "TStrings" öffnet sich der String List Editor, wie er auf der rechten Seite zu sehen ist. Hier kann der Text der Infobox bearbeitet werden.

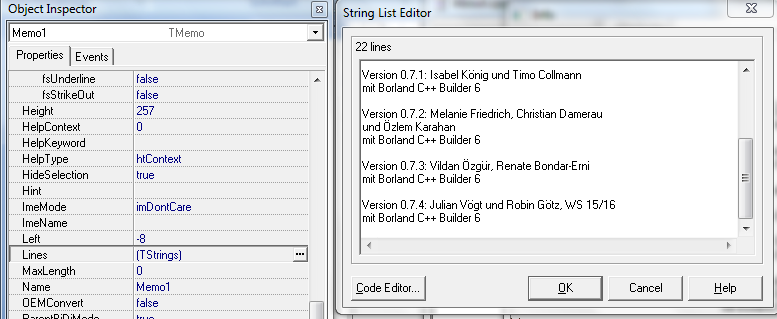


Abbildung 5 - Änderung der Infobox

# Zusatzaufgaben

Zusätzlich zu den Aufgaben in der Projektbeschreibung wurden zwei weitere Zusatzaufgaben genannt. Es sollte versucht werden diese ebenfalls zu lösen. Sollte eine Lösung nicht möglich sein, so sollen Gründe benannt werden warum die Aufgaben nicht lösbar sind und, falls vorhanden, mögliche Lösungswege erläutert werden.

## Einbinden des GLPK-Solvers

Die erste Zusatzaufgabe bestand aus der Einbindung eines weiteren Solvers (GLPK) in Power-LP. Jedoch konnte dies aus den folgenden Gründen nicht umgesetzt werden.

* Die Programmiersprache C++ wird in der Studienzeit an der HTWG nicht gelehrt. Dazu kommt, dass diese deutlich komplexer ist als die bisher gelernte Programmiersprache JAVA. Eine Einarbeitung in C++ war in der vorhandenen Zeit nur bedingt möglich. Daher war es nicht möglich die benötigten Änderungen am Code vorzunehmen.
* In der beim Solver GLPK mitgelieferten Dokumentation ist nicht beschrieben wie der Solver in eine bestehende Software einzubinden ist.
* Die Einbindung eines Solvers in Power-LP ist sehr komplex. Die Source-Code Struktur beinhaltet viele Abhängigkeiten die Bedacht werden müssen. Die Kombination aus fehlender Dokumentation, fehlenden Programmierkenntnissen und komplexen Source-Code machten eine Einbindung der GLKP-Solvers in der vorgegebenen Zeit und Gruppengröße unmöglich.

Zur Umsetzung dieser Aufgabe wird daher ein Teamprojekt oder ALO-Projekt mit höherem Personaleinsatz empfohlen.

## Korrektes Ausführen von Power-LP-Dateien

Die zweite Zusatzaufgabe betrifft das direkte Öffnen von Power-LP-Dateien. Diese können die Formate „.lpi“, „.lp“ und „.mps“ haben.

Macht man einen Doppelklick auf eine Word-Datei, so wird automatisch das Textverarbeitungsprogramm Word geöffnet. Der Inhalt der ausgewählten Datei wird in Word direkt angezeigt. Will man das Gleiche mit einer Power-LP-Datei machen, so muss man zunächst selbst das Programm auswählen mit dem die Datei geöffnet werden soll. Diese Information wird von Windows gespeichert, sodass die Datei in Zukunft direkt mit einem Doppelklick in Power-LP geöffnet wird. Hierbei wird jedoch nur Power-LP geöffnet, so, als würde man die Anwendung "normal" starten. Der Inhalt der Datei, wird nicht in Power-LP angezeigt. Die Zusatzaufgabe war, dies zu ändern.

Eine langwierige Recherche und Codeanalyse gibt uns die Möglichkeit theoretische Lösungsansätze zu formulieren. Aufgrund der hohen Komplexität des Power-LP Sourcecodes, Programmierung in C++ und der späten Äußerung des Wunsches zum Einbaus dieses Features, dass nicht im Pflichtenheft vereinbart war, war es jedoch zeitlich nicht mehr möglich dies umzusetzen. Folgend die theoretischen Lösungsansätze für folgende ALO / Teamprojekte:

Wird Power-LP "normal" geöffnet, wird der untenstehende Code in der Datei main.cpp aufgerufen:

TMDIChild\* \_\_fastcall TMainForm::CreateMDIChild(String pName) {

TMDIChild \*Child;

AnsiString Name = pName;

if (Name.Length() == 0) {

Name = "NEU";

Name += IntToStr(MDIChildCount + 1);

}

Child = new TMDIChild(Application);

Child->Caption = Name;

if (FileExists (Name)) Child->LoadFile(Name.c\_str());

return Child;

}

Es wird überprüft, ob der Funktion CreateMDIChild ein Parameter übergeben wurde. Ist dies nicht der Fall wird ein neues File mit dem Namen "Neu1" erstellt. Öffnet man eine Power-LP-Datei, wird genau das gleiche gemacht, es wird der gleiche Code ausgeführt. Jedoch sollte erkannt werden, dass eine bestehende Datei ausgeführt wird und deren Werte eingelesen werden.

Die Funktion FileOpen1Execute macht genau dies. Es wird ein Fenster geöffnet, in dem man eine Datei auswählen kann. Anschließend wird die Datei in Power-LP angezeigt. Als Name wird der Dateipfad angezeigt.

void \_\_fastcall TMainForm::FileOpen1Execute(TObject \*Sender) {

OpenDialog->InitialDir = reg\_getdatadir();

if (OpenDialog->Execute()) {

char dir[200];

getcwd(dir,sizeof(dir));

reg\_setdatadir(dir);

CreateMDIChild(OpenDialog->FileName);

}

}

Es wurde versucht diese zwei Funktionen miteinander zu verknüpfen, jedoch ohne Erfolg. Auch verschiedene Code-Beispiele aus dem Internet halfen nicht das Problem zu lösen.

Ein Ansatz war folgender: Wird eine Datei in Windows geöffnet, so werden verschiedene Parameter mitgegeben, wie z.B. der Dateipfad und der Name. Man müsste in Power-LP diese Parameter auslesen und anschließend Power-LP sagen, dass diese Datei nun geöffnet werden soll.

# Probleme und Einschränkungen der Software

Pfade

Große Probleme bereiteten Fehlermeldungen die anfangs nicht einzuordnen waren. Nach langer Analyse und Testing stellt sich heraus, dass der Solver „Weidenauer Optimizer“ nur mit Pfaden zurechtkommt, dessen Zeichenanzahl unter 80 liegt. Jedoch ist dies sehr unregelmäßig. In verschiedenen Tests konnten auch Pfade mit ca. 100 Zeichen ausgeführt werden. Ausgiebiges Testen bestätigte jedoch, dass ein zu langer Pfad zu Fehlern führt. Es wird eine Fehlermeldung generiert und der Weidenauer-Solver funktioniert nicht. Das Problem ließ sich nicht lösen, da der Weidenauer-Solver selbst in seinem Programmcode festsetzt wie lange die Pfade sein dürfen. Die Software ist sehr alt und in Pascale geschrieben. Eventuell kann ein kommendes ALO-Projekt dieses Problem in der Zukunft lösen. Eine Ausführung aller anderen Solver war trotz der Fehlfunktion des Weidenauers bei langen Pfaden trotzdem möglich.

Auch der Solver XA bereitet Probleme mit Pfaden. Werden Umlaute wie Ö,Ä oder Ü im Pfad eingesetzt oder enthält der Pfad Leerzeichen ist eine Ausführung der Solver nicht mehr möglich. Dies liegt ebenfalls nicht an unserem Power-LP-Sourcecode sondern am Programmcode der besagten Solver. Dieser Umstand kann ebenfalls in einem kommenden ALO-Projekt gelöst werden.