The Network-Enabled Optimization System (NEOS) Server

Joseph Czyzyk, Michael P. Mesnier, and Jorge J. More

CRPC-TR96707-S
Revised March 1997

Center for Research on Parallel Computation
Rice University
6100 South Main Street
CRPC - MS 41
Houston, TX 77005

Submitted in November 1997; Also available as Argonne National Laboratory Preprint MCS-P615-1096

ARGONNE NATIONAL LABORATORY
9700 South Cass Avenue
Argonne, Illinois 60439

THE NETWORK-ENABLED OPTIMIZATION SYSTEM (NEOS) SERVER

Joseph Czyzyk, Michael P. Mesnier, and Jorge J. Moré

Mathematics and Computer Science Division

Preprint MCS-P615-1096

October 1996

Revised Version (March 1997)

This work was supported by the Mathematical, Information, and Computational Sciences Division subprogram of the Office of Computational and Technology Research, U.S. Department of Energy, under Contract W-31-109-Eng-38, by a grant of Northwestern University to the Optimization Technology Center, and by the National Science Foundation, through the Center for Research on Parallel Computation, under Cooperative Agreement No. CCR-9120008.

THE NETWORK-ENABLED OPTIMIZATION SYSTEM (NEOS) SERVER*

Joseph Czyzyk, Michael P. Mesnier, Jorge J. Moré

Abstract

The Network-Enabled Optimization System (NEOS) is an environment for solving optimization problems over the Internet. Users submit optimization problems to the NEOS Server via e-mail, the World Wide Web, or the NEOS Submission Tool. The NEOS Server locates the appropriate optimization solver, computes all additional information (for example, derivatives and sparsity patterns) required by the solver, links the optimization problem with the solver, and returns a solution. This article discusses the design and implementation of the NEOS Server.

1 Introduction

The Network-Enabled Optimization System (NEOS) is an Internet-based service for optimization. The goal of NEOS is to be the definitive site for optimization information and technology, providing users not only with up-to-date literature on optimization but ready access to a growing library of optimization software. The main components of NEOS are the NEOS Guide and the NEOS Server. The NEOS Guide is a Web-based guide to optimization theory and practice, while the NEOS Server is an Internet-based client/server application that provides access to a library of optimization software. This article focuses on the NEOS Server.

The NEOS Server introduces a novel approach to the solution of optimization problems. In the conventional approach the user must first identify and obtain the appropriate piece of optimization software; write code to define the problem in the manner required; and then compile, link, and run the software. Typically, Fortran or C code must be written to define the problem, compute function values and derivatives, and specify sparsity patterns. With NEOS, users are able to solve optimization problems over the Internet by providing only the minimal input required to specify the problem.

The NEOS Server handles linear and nonlinear optimization problems, ranging from linear programming to nonlinearly constrained optimization; we are in the process of adding solvers to handle optimization problems subject to integer variables. We illustrate the

*Mathematics and Computer Science Division, Argonne National Laboratory, Argonne, IL 60439-4844. This work was supported by the Mathematical, Information, and Computational Sciences Division subprogram of the Office of Computational and Technology Research, U.S. Department of Energy, under Contract W-31-109-Eng-38, by a grant of Northwestern University to the Optimization Technology Center, and by the National Science Foundation, through the Center for Research on Parallel Computation, under Cooperative Agreement No. CCR-9120008.

advantages of the NEOS way of solving optimization problems with the general nonlinearly constrained optimization problem

$$\min \{f(x) : x_l \leq x \leq x_u,\ c_l \leq c(x) \leq c_u\}. \tag{1.1}$$

This formulation requires that we determine variables $x \in \mathbb{R}^n$ that minimize the function $f : \mathbb{R}^n \mapsto \mathbb{R}$ subject to the bounds $x_l \leq x \leq x_u$ and the nonlinear constraints $c_l \leq c(x) \leq c_u$ specified by a mapping $c : \mathbb{R}^n \mapsto \mathbb{R}^m$.

The conventional approach to the solution of (1.1) requires that the user provide additional information besides the function f, constraint function c, the bounds x_l, x_u on the variables, and bounds c_l, c_u on the constraints. In particular, the user is often asked to provide a subroutine to evaluate the gradient $\nabla f(x)$ of f and the Jacobian matrix $c'(x)$ of c for any trial x generated by the optimization software. In addition, for sparse problems, the user may also be required to supply the sparsity patterns of the Hessian matrix $\nabla^2 f(x)$ and the Jacobian matrix $c'(x)$.

The conventional user of optimization software must also select and retrieve the appropriate optimization software and link this software with the code to evaluate the function and constraints. In general, the solution process for the nonlinearly constrained problem (1.1) requires the following steps:

Decide on the appropriate optimization solver for the application.
Retrieve the optimization solver.
Develop code to evaluate the functions f and c, and the bounds x_l, x_u and c_l, c_u.
Develop code for the gradient, Hessians, and sparsity patterns.
Develop code to link the optimization solver with the application.
Debug, compile, and execute.
Interpret the results.

The NEOS Server provides a novel alternative to this process. For the constrained optimization problem (1.1), the user submits the problem dimensions m and n, subroutines to evaluate the function f and c, the bounds x_l, x_u, c_l, c_u, and the starting point. The NEOS Server locates the appropriate solver; computes the gradient of f; the Jacobian matrix of c; the sparsity patterns of the Hessian of f and of the Jacobian matrix of c; and links with the solver. The user is given a solution vector, along with run-time statistics. Most of the computation is done in the background, hidden from the user.

Derivatives (gradients, Jacobian and Hessian matrices) of nonlinear problems and the sparsity patterns are determined by ADIFOR/SparsLinc [3] for Fortran codes and by ADOL-C [9] for C codes. Execution of the codes takes place on workstations provided by the software administrators. A variety of machines, even a massively parallel processor, could be used to solve the problem; the only restriction is that the workstation must

run UNIX with support for TCP/IP. At present these workstations reside at Argonne National Laboratory, Northwestern University, and the University of Wisconsin. The NEOS solver [8] for complementarity problems at Wisconsin is of special interest since, in addition to the scheduling facilities already provided by the Server, the Condor [10] scheduling system provides site-specific scheduling on clusters of workstations.

We have concentrated on the nonlinearly constrained optimization problem (1.1), but the NEOS Server is able to solve a wide range of optimization problems. At present NEOS supports solvers in the following areas:

Unconstrained optimization
Bound constrained optimization
Nonlinearly constrained optimization
Complementarity problems
Linear network optimization
Linear programming
Stochastic linear programming

Users with optimization problems in these areas are able to use the latest version of state-of-the art optimization software. Developers of optimization software, on the other hand, are provided with a potentially large user base and immediate feedback on their software.

Since the NEOS Server simplifies the solution process, users are able to focus their attention on solving the optimization problem, rather than on the details of the optimization software. In particular, users are able to solve the problem with several algorithms or with different formulations of the problem. For example, a user with an unconstrained optimization problem

$$\min \{f(x) : x \in \mathbb{R}^n\}$$

can solve this problem with any of the solvers in the unconstrained optimization area, or may add bounds to the problem and solve the problem as the bound-constrained problem

$$\min \{f(x) : x_l \leq x \leq x_u\}.$$

The only change needed is to write a subroutine to compute the bounds x_l and x_u.

As another example, users with a linear programming problem can model the problem as a stochastic linear programming problem by providing additional parameters that specify the randomness of a given model. Even though solving a stochastic linear program can be considerably different from a standard linear program, the user is shielded from these details by the NEOS Server.

The NEOS Server grew out of an early, e-mail-based system for the submission of linear algebra problems. The beta version of the Server was released in May 1995; the Server was

made available for general use in September 1995. Projects with similar aims to NEOS are in the planning stages. The NetSolve [5] and RCS [1] projects, for example, are intended to allow users to build an application by calling libraries with remote procedure calls. The current interfaces of NetSolve and RCS deal only with linear algebra problems. Moreover, neither NetSolve nor RCS is currently available for general use.

We are planning to extend NEOS by allowing remote procedure calls. This extension would enable users to call NEOS directly from their Fortran and C programs. Extensions of NEOS to other related areas are also being considered.

This article concentrates on two important aspects of the design and implementation of the Server: user interfaces and software registration. The usefulness of the Server is dictated, to a large extent, by the interfaces used to submit optimization problems. Consequently, submission of optimization problems must be intuitive and simple. These issues are discussed in Section 3. The Server should also be able to grow and add optimization solvers. Since the optimization solvers can be provided by a wide range of optimization experts, the registration of a new solver with the Server must be automatic and simple, with little intervention by the Server administrators. The software registration process is discussed in Section 4. We conclude the article with a brief introduction to the solvers available in the NEOS Software Library.

2 Design of the NEOS Server

The basic goal of the Server is to provide Internet access to a growing library of optimization software—the NEOS Software Library—with user interfaces that abstract the user from the optimization software. The user needs only to provide a definition of the problem with the required data.

Figure 2.1 shows how the NEOS Server interacts with Internet users. In the typical situation, a user submits a problem to the NEOS Server via one of the three interfaces. The Server then locates the appropriate solver in the library and performs a file transfer with all the user's data. The software administrator for the solver has written code that checks the input for consistency, generates any additional information that may be necessary, executes the optimization solver, and generates the appropriate results. NEOS then returns the solution to the user.

Providing Internet access to a growing library of optimization solvers requires a design with a range of capabilities:

> The Server requires only a minimal specification of the optimization problem: Server interfaces either piggy-back on existing media (e-mail and the World Wide Web) or can be easily downloaded and installed on the client machine (the NEOS Submission Tool).

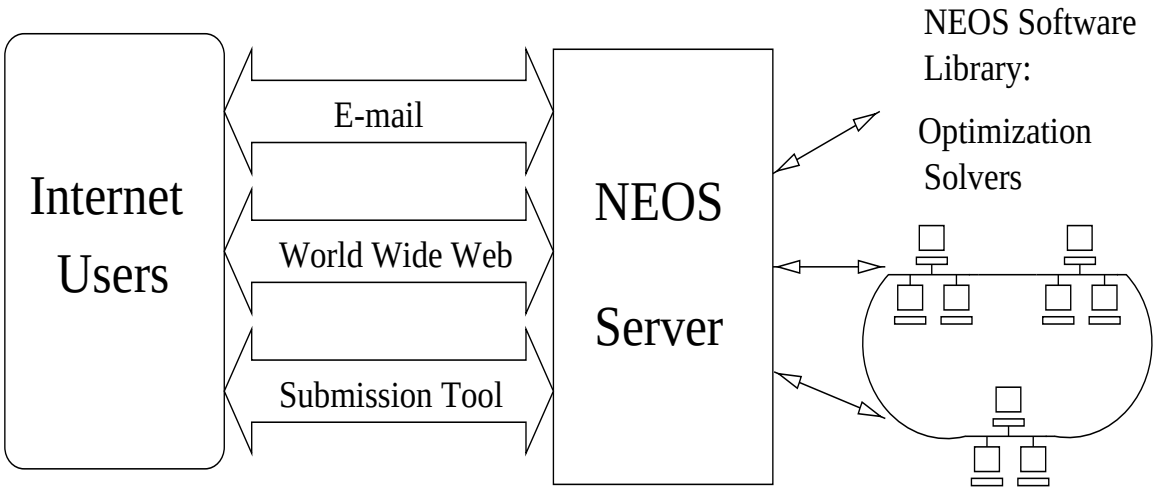

Figure 2.1: NEOS Server: Interaction with Internet Users

The Server provides a uniform, automated method for software registration. The Server manages the library of optimization solvers that have been registered with the NEOS Server.

The Server is fault-tolerant. In particular, the Server automatically restarts itself and re-establishes connections to client machines. In addition, the solvers for the optimization software packages have been written to anticipate a variety of errors and to provide useful feedback to users.

The Server maintains a logging system of all NEOS transactions. NEOS administrators have a searchable client database.

In the remainder of the article we elaborate on these capabilities. We concentrate, in particular, on the user interfaces and the registration process used by the NEOS Server.

3 The User Interfaces

The NEOS Server gives Internet users the choice of three interfaces: e-mail, the World Wide Web, and the NEOS Submission Tool. In all cases, a user with an optimization problem selects the appropriate solver from the list of available solvers, composes the job submission, and submits the problem to NEOS. The user interfaces are designed so that this process is intuitive and requires the minimal amount of information. Once received by NEOS, job submissions are parsed to determine the problem type and then scheduled on an appropriate workstation. The results are returned to the user through the same interface that was used to submit the problem.

3.1 E-Mail

The e-mail interface is the most primitive form of job submission and is used primarily by users who do not have direct Internet access or do not operate in a windowing environment such as X Windows or Windows 95. General information about the Server can be obtained by mailing the message *help* to `neos@mcs.anl.gov`. The user receives the general help file with a short description of the Server and a list of the solvers available in the NEOS Software Library. The user can obtain additional information on a specific solver by sending NEOS the message

```
type   <problem type identifier>
solver <solver identifier>
help
```

where `<problem type identifier>` is replaced by the identifying token for the type of optimization problem and `<solver identifier>` is replaced by the token for the optimization solver; a complete list of NEOS token identifiers is returned with the general help file.

The help file for each solver includes information on how to specify the optimization problem, a template for submitting problems, and sample job submissions. Users of the e-mail interface must place the specification of the optimization problem in a file, delimiting this data with a set of predefined tokens. A discussion of these tokens is included with the information on the selected software. For example, the template

```
type   UCO
solver NMTR

n = <number of variables>

begin.initpt
  <fortran code>
end.initpt

begin.fcn
  <fortran code>
end.fcn
```

can be used to submit unconstrained optimization problems to NEOS. The first line tells NEOS the type of optimization problem that is being submitted. In this case, the type is UCO (UnConstrained Optimization). The second line specifies that the user wants to use the NMTR solver. The number of variables is specified by setting `n` to the appropriate value. The initial point subroutine is placed between the `begin.initpt` and `end.initpt` tokens, while the function evaluation subroutine must appear between the `begin.fcn` and `end.fcn` tokens.

```
subroutine fcn(n,x,f)
integer n
double precision f
double precision x(n)

integer nx, ny
double precision lambda
lambda = 0.008
nx = 10
ny = 10
call dodc(nx,ny,x,f,lambda)
end

subroutine dodc(nx,ny,x,f,lambda)
.....
end
```

Figure 3.1: Example of Fortran subroutine for the NEOS Server

The submission format requires that the user's function be in the format required by NEOS. For example, NEOS requires that the calling sequence of the function subroutine be of the form `fcn(n,x,f)`. Users with a different calling sequence can write a wrapper routine. For example, if the user is using a function with a calling sequence of `dodc(nx,ny,x,f,lambda)`, where `nx*ny` are the number of variables, and `lambda` is a parameter, then the routine shown in Figure 3.1 is suitable. Clearly, similar routines can be written for a wide variety of optimization problems.

The main disadvantage of the e-mail interface is that the user needs to provide the specification of the optimization problem with the required tokens. This can lead to unexpected errors. For example, many mailers attach signatures to each mail message. The use of the NEOS token `END-NEOS-INPUT` prevents NEOS from reading further in the message.

3.2 World Wide Web

Submission via the World Wide Web makes job submission easy for users with access to Netscape or a similar browser. The NEOS Web interface begins with the URL

`http://www.mcs.anl.gov/home/otc/Server/`

for the NEOS Server. This homepage describes the NEOS Server and lists the available solvers within the Software Library. By following the appropriate links, the user can obtain additional information on a solver and submit a problem to the Server.

Unlike e-mail users, Web users need not be concerned with tokens and data delimitation. Instead, Web users specify the optimization problem via radio/check buttons, text entries, and URL addresses.

An important advantage of the Web interface is that the user gets immediate feedback on the job submission. Error messages are returned so that the user is able to make corrections and resubmit the job. The main disadvantage of the Web interface is that the job submission must be placed in Web-accessible files. For example, instead of placing the function evaluation subroutine in an e-mail message with the appropriate tokens, the user must place the function evaluation subroutine in a Web-accessible file. An anonymous ftp site is an example of a Web-accessible file.

The main advantage of the Web interface is that additional information on the solver or on the optimization theory behind the solvers can be readily provided. In particular, each solver has a link that points to the appropriate entry in the NEOS Guide.

3.3 Submission Tool

The NEOS Submission Tool is designed for users with a direct Internet connection. This is the fastest of the three interfaces because communication is done with TCP/IP sockets. Like the Web interface, the Submission Tool provides appropriate entries for the users' data. Unlike the Web interface, the Submission Tool allows the uploading of files from the user's local file space to the NEOS Server.

The NEOS Submission Tool is a TCl/Tk [11] application. Users can obtain more information and download the submission tool from the URL

`http://www.mcs.anl.gov/home/otc/Server/neos/subtool.html`

Although the Submission Tool is written with TCl/Tk, users need only have Perl [12] installed.

Figure 3.2 shows the NEOS Submission Tool form for the NMTR solver for unconstrained optimization. The user needs to specify only the language used in the job submission, the number of variables, and the files for the initial point and function evaluation subroutines. *Browse* buttons are available to ease the specification of the various files.

As with the e-mail and Web interfaces, the Submission Tool comes with a description of the Software Library and sample submissions.

4 Software Registration

The NEOS Server is able to expand through software registration, a simple, automated process that allows additional optimization solvers to be registered with NEOS. Registered software administrators must provide the software and the hardware; all client/server services are provided by NEOS. The following information is required for registration:

The type of optimization problem (for example, nonlinearly constrained optimization)
The name of the solver (for example, LANCELOT)

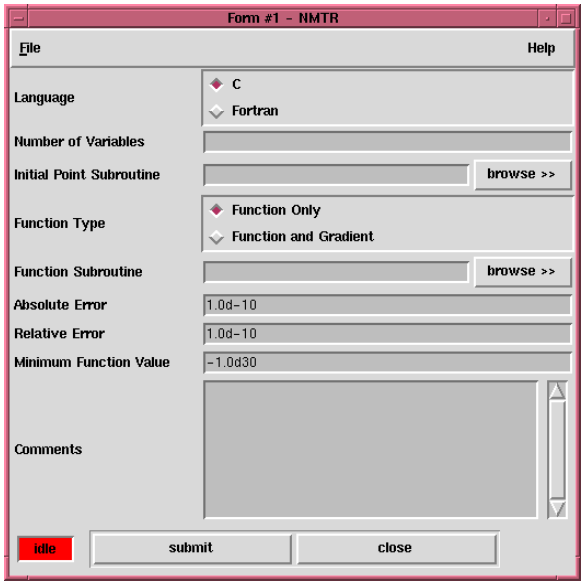

Figure 3.2: The NEOS Submission Tool

The e-mail address of the software administrator(s) and a password
Addresses for registered workstations
A configuration file
Help files

The type of the optimization problem and the name of the solver are the identifiers that are also used during e-mail submission. For example, registration of NMTR for unconstrained minimization was done by setting the type to `UCO` and setting the name of the solver to `NMTR`. Similarly, registration of LANCELOT would be done by setting type to `NCO` (nonlinearly constrained optimization) and setting the name of the solver to `LANCELOT`.

The list of workstations must contain addresses for any UNIX machine that supports TCP/IP sockets and that can execute the solver.

The configuration file and the help files are used to generate the e-mail, Web, and Submission Tool interfaces. This process is entirely automated, and interfaces are generated to be consistent with other solvers in the Server.

After registration, the software administrator must download and install the NEOS Communications Package. This package contains a client/server application, `local_server`, that allows a client (the NEOS Server) to connect and request that a job be executed. Figure 4.1 shows the information required by the NEOS Communications Package.

The software identifier and password were provided by the software administration during the registration process. The only new piece of information required by the NEOS Communications Package is the path of the solve script.

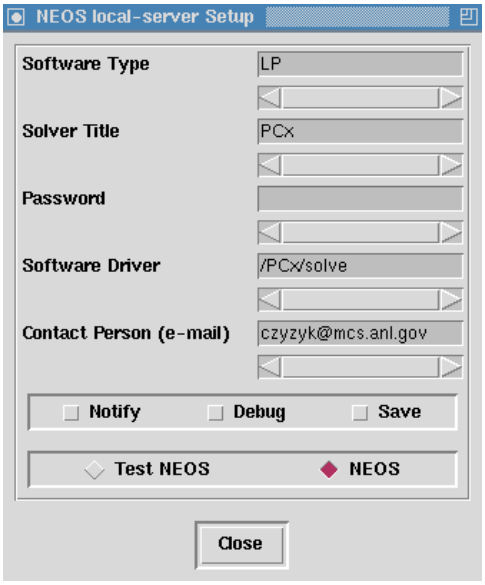

Figure 4.1: NEOS Communication Package

The `local_server` executes as a daemon process on the registered workstations. Once started, the `local_server` binds to the first available port on the list of registered workstations and waits for connections from NEOS. When an optimization problem is received by the NEOS Server, the Server contacts the first available workstation. Upon connection, the `local_server` downloads the user's data, launches the optimization solver, and returns the results generated by the solver. During this entire process the connection between the user and the remote software package is maintained—with NEOS as the liaison. This allows software packages to communicate with the user.

Facilities exist for automatically restarting `local_server` in the event that the host machine is rebooted or `local_server` is killed.

We illustrate the solution process with a nonlinearly constrained optimization problem submitted to the LANCELOT solver. The LANCELOT solver is located somewhere on the Internet, but only the Server is concerned with the actual location. Figure 4.2 shows the sequence of events during the solution process.

The user submits a nonlinearly constrained optimization problem with the NEOS Submission Tool. In this case the supplied code is in Fortran.

The NEOS Server contacts the first available workstation.

The `local_server` performs a file transfer from the NEOS Server. The transfer includes all of the user's data.

The `local_server` launches the solver for the problem type. The solver checks the data, compiles the user's subroutines, generates gradients and Jacobian matrices with

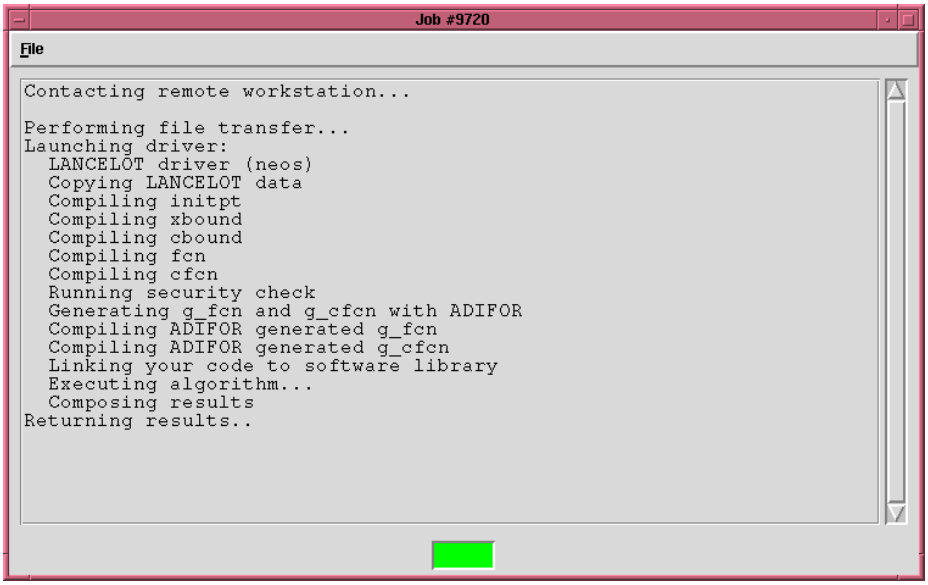

Figure 4.2: Output from the NEOS Submission Tool

ADIFOR, links with the appropriate libraries, executes the optimization software, and generates a solution.

The `local_server` returns the job results to the NEOS Server.

A similar process occurs for every NEOS job, that is, data is transferred from the user, to the NEOS Server, and then to a remote workstation for execution. The connection between the user and the software package is maintained during this process, thus allowing for a real-time report of the software's progress. Both the Submission Tool and Web interfaces support this status capability; e-mail users receive only the end result.

5 The NEOS Software Library

The Software Library is the set of optimization solvers that have been registered with the NEOS Server. Participating hosts have chosen to make their software available to the Internet via NEOS. Not only is the user relieved of downloading, compiling, and installing the software, but the individual software administrators are relieved of software distribution.

The NEOS Software Library is distributed. In other words, the actual code for each optimization solver may reside anywhere on the Internet. Communication between the NEOS Server and the remote workstations where the software is located is handled by the NEOS Communications Package—an Internet client/server application providing socket communications between the NEOS Server and the remote software.

The NEOS Server has solvers in seven different areas of optimization. Solvers in other areas (for example, systems of nonlinear equations and nonlinear least squares) will be

added in the near future. Our current offering consists of the following:

Unconstrained optimization: NMTR
Bound constrained optimization: L-BFGS-B, LANCELOT
Nonlinearly constrained optimization: LANCELOT
Complementarity problems: PATH
Linear network optimization: NETFLO, RELAX4
Linear programming: AUGMENTED, PCx
Stochastic linear programming: AUGMENTED, MSLIP

We do not describe all the optimization solvers currently in the Software Library since they are listed in the NEOS Web interface, and have pointers to the relevant literature. Additional information on optimization can be obtained from the URL

`http://www.mcs.anl.gov/home/otc/Guide/`

for the NEOS Guide. The main issue that needs to be addressed is the input format for the specification of the problem.

The nonlinear solvers accept Fortran input, but most of the solvers also accept C input. For example, the L-BFGS-B solver [13] for bound constrained optimization problems accepts input in terms of either Fortran or C code. On the other hand, the LANCELOT [6] solver for general constrained optimization problems accepts the problem specification in terms of Fortran subroutines or in the SIF format.

For nonlinear solvers capable of solving large-scale problems, we require that the solver accept functions in partially separable form, that is, functions of the form

$$f(x) = \sum_{i=1}^{m} f_i(x),$$

where each element function f_i depends on only a few components of x, and m is the number of element functions. Both L-BFGS-B and LANCELOT follow this requirement, which allows [4] the efficient generation of gradients with automatic differentiation techniques.

The range of formats used to specify problems for linear solvers is strongly dependent on the area. Instead of Fortran or C codes, these solvers accept the input as a data file. The linear programming solvers (for example, PCx [7]) accept the problem specification in the MPS format, although other formats are clearly possible. Solving stochastic linear programming problems with the NEOS Server is straightforward because solvers in this area accept the input in a format based on the MPS standard and designed to promote the efficient conversion of originally deterministic problems.

Solvers in other areas of the NEOS Server use formats that have been adopted in these areas. Thus, solvers for (linear) minimum-cost network flow problems (for example, RELAX-IV [2]) accept input in the DIMACS format.

Acknowledgments

The design of the NEOS Server was developed in consultation with members of the Optimization Technology Center, in particular, Jorge Nocedal, Steve Wright, Richard Marynowski, Jon Owen, Emmett Tomai, and Andrew Crane. Other contributors include Bill Gropp, for the original design of the e-mail based component of the Server, and the developers of ADIFOR and ADOL-C. We would also like to thank those who have provided valuable feedback during the development and testing of our software registration system, in particular, Ali Bouaricha, Michael Ferris, Zhijun Wu, and Ciyou Zhu.

References

[1] P. Arbenz, W. Gander, and M. Oettl, *The remote computation system*, in High-Performance Computing and Networking, H. Liddell, A. Colbrook, B. Hertzberger, and P. Sloot, eds., no. 1067 in Lecture Notes in Computer Science, Springer-Verlag, Berlin, 1996, pp. 820–825.

[2] D. Bertsekas and P. Tseng, *RELAX-IV: A faster version of the RELAX code for solving minimum cost flow problems*, Technical Report, Massachusetts Institute of Technology, Cambridge, Massachusetts, 1994. Also available by ftp from lids.mit.edu in /pub/bertsekas/RELAX/RELAX4.PS.

[3] C. Bischof, A. Carle, P. Khademi, and A. Mauer, *The ADIFOR 2.0 system for the automatic differentiation of Fortran 77 programs*, Preprint MCS-P381-1194, Argonne National Laboratory, Argonne, Illinois, 1994. Also available as CRPC-TR94491, Center for Research on Parallel Computation, Rice University.

[4] A. Bouaricha and J. J. Moré, *Impact of partial separability on large-scale optimization*, Comp. Optim. Appl., 7 (1997), pp. 27–40.

[5] H. Casanova and J. Dongarra, *NetSolve: A network server for solving computational science problems*, Technical Report CS-95-313, University of Tennessee, Knoxville, Tennessee, 1995.

[6] A. R. Conn, N. I. M. Gould, and P. L. Toint, LANCELOT, Springer Series in Computational Mathematics, Springer-Verlag, 1992.

[7] J. Czyzyk, S. Mehrotra, and S. J. Wright, *PCx user guide*, Technical Memorandum ANL/MCS-TM-217, Argonne National Laboratory, Argonne, Illinois, 1996. Also available as ftp://ftp.mcs.anl.gov/pub/neos/PCx/PCx-user.ps.

[8] M. C. Ferris, M. P. Mesnier, and J. J. Moré, *The NEOS Server for complementarity problems: PATH*, Technical Report 96-08, University of Wisconsin, Madison,

Wisconsin, 1996. Also available as MCS-P616-1096, Mathematics and Computer Science Division, Argonne National Laboratory.

[9] A. Griewank, D. Juedes, and J. Utke, *ADOL-C: A package for the automatic differentiation of algorithms written in C/C++*, ACM Trans. Math. Software, 22 (1996), pp. 131–167.

[10] M. J. Litzkow, M. Livny, and M. W. Mutka, *Condor - A hunter of idle workstations*, in Proceedings of the 8th International Conference on Distributed Computing Systems, Washington, District of Columbia, 1988, IEEE Computer Society Press, pp. 108–111.

[11] J. K. Ousterhout, *Tcl and the Tk Toolkit*, Addison-Wesley, 1994.

[12] L. Wall, T. Christiansen, and R. L. Schwartz, *Programming Perl*, O'Reilly & Associates, Inc., second ed., 1996.

[13] C. Zhu, R. H. Byrd, P. Lu, and J. Nocedal, *L-BFGS-B: Fortran subroutines for large-scale bound constrained optimization*, Technical Report, Northwestern University, 1994.