ISSN:2231-0711
www.ijcset.net

Non Linear Programming Computation Outsourcing in the Cloud

Bhavsingh Maloth, Juveriah Suman, G.Saritha, A.Chandrasekhar

Dept of Information Technology,
G.Pullaiah College of Engineering & Technology, Kurnool, Andhra Pradesh, India
bhavsinghm@gmail.com

***Abstract*—Cloud Computing is termed with great potential in providing robust computational power to the society at low cost. This enables customers with limited computational resources to outsource the large computation workloads to the cloud, and to enjoy massive computational power, bandwidth, and storage economically. Connecting the cloud to an intrinsically insecure computing platform from the viewpoint of the cloud customers, must model mechanisms not only protect sensitive information by enabling computations with secured encrypted data, but also protect customers against malicious behaviors by including the validation of the computation result. Such a functionality of general secure computation outsourcing was shown to be feasible in theory recently, but to design mechanisms that are practical efficient remains a very challenging problem. Nonlinear programming problems are in more general difficult than linear programming problems, and often hence the way out found is only a local optimum. The way out methods for nonlinear programming models varies, which can result to different nonlinear solvers giving different local optima for to same problem.**

This application with multiple objective optimization techniques based on the methods of nonlinear goal programming in performing optimal synthesis of general planar mechanisms is presented. This optimization technique and the objectives of the design problem are first identified and prioritized according to their relative importance value. The mechanism analysis is performed to identify the design variables and the relationships to the dependent variables. The nonlinear goal programming technique is exploitd to determine the optimal values for the design variables that best satisfy the desired objectives of the problem. Uniqueness of this optimization technique lies in its ability to include all its objectives directly in the optimization process. So this method eliminates the need to identify a single objective for the purpose of optimization. Multiple illustrative mechanism synthesis problems with multiple objectives studied for optimal way outs using the nonlinear goal programming technique and the method produced accurate and acceptable way outs. This application of method to three typical industrial type mechanism synthesis problems is presented.

Keywords: Linear Programming, Non Linear Programing,Affine mapping,Duality theorem ,

INTRODUCTION

Focusing to engineering computing and optimization tasks, this paper examines secure outsourcing of widely applicable linear programming (LP) computations.To achieve practical efficiency, this mechanism design explicitly decomposes our LP computation outsourcing into public LP solvers on the cloud and private LP parameters owned by the customer. This results in flexibility allows to explore appropriate security/efficiency tradeoff via higher-level abstraction of LP computations compared to the general circuit representation. Particularly,formulating private data owned by the customer for LP problem as a set of matrices and vectors, which able to develop a set of efficient privacy-preserving problem transformation techniques, which allows to transform original LP problem into some arbitrary one In the time of protecting sensitive input/output information. During the validation of the computation result, we further explore the fundamental duality theorem of LP computation and to derive the necessary with sufficient conditions which corrects the result must be satisfied. This result verification mechanism termed extremely efficient and it incurs close-to-zero additional cost on both cloud server and customers. Extensive security analysis and the experiment result shows the immediate practical ability of our mechanism design.

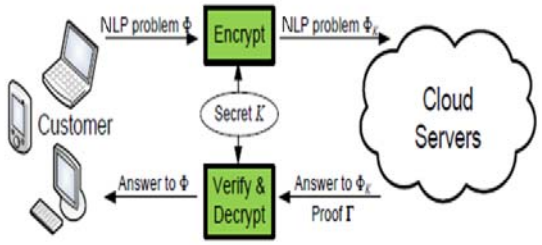

Fig. 1: Architecture of secure outsourcing non-linear programming problems in Cloud Computing

Considering the linear program where the cost coeffcients calculated as uncertain, for which we have a Bayesian prior.As we can collect the information to improve the understanding of these coeffcients, but it may be expensive, giving a separate optimizing problem and the collection of information for improving the quality of the way out relative to the true cost coeffcients. Hence formulating this information collection of the problem for linear programs for the first time, and to derive knowledge gradient policy that maximizes the marginal value of each measurement. We prove this policy is asymptotically optimal, and demonstrates its performance of our algorithm on a network flow problem.
Consider the standard form of a linear program, given by

$V(c)=\max_{X} c^{T}x$

s.t. $Ax=b$

$x \geq 0$

and suppose that this vector c of objective co-efficients is unknown. we have certain initial information on the problem that allows to construct a Bayesian prior around c. hence, view c as a random variable whose distribution represents uncertainty in the beliefs. We have the ability for making the noisy measurements of individual co-efficient of c. Every measurement provides new information which can be exploited to update and improve the beliefs about objective

co-efficient. Measurements are assumed as time-consuming and expensive; choosing any co-effcient to measure at any time, but the total budget allocated for measurements is finite. Moreover, our beliefs about different co-effcients may be correlated, meaning that the single measurement could potentially provide exploitful information about more than one component of c. This problem is to determine that co-effcients to measure in order to come as close as possible to the optimal way out of the true LP

A basic idea for a linear program with stochastic parameters has been around for many years. Thus this problem was originally formulated by Danzig (1995) and this proposed approach in the early study was to solve a deterministic LP where c was replaced with its mean. Problem for finding IEV (c), expected optimal value for the linear program, was approached by Madansky (1960) and Itami(1974) . This study is considered for various theoretical properties of the problem, but computing on the desired expectation still posed a challenge. Assumptions and approximations for the multi-stage version of this problem were considered by Birge (1982) . Multi-stage stochastic problems are highly studied in the area of stochastic programming;. Moreover, the analysis will give insights on how a stronger mechanism should be designed. Nonlinear programming (NLP) is the route of solving a system of equalities and inequalities, communally termed constraints, on a set of unknown real variables, along with the objective function to be maximized or minimized, where some of the constraints or the objective function are nonlinear.

A distinctive non convex problem is that of optimising hauling costs by selection from a set of transportion methods, one or more of which put on view economies of scale, with a range of connectivity's and capacity constraints. Considering an illustration would be petroleum product transport given a assortment or combination of pipeline, rail tanker, road tanker, river barge, or coastal tankship. Outstanding to economic batch size the cost functions may have discontinuities in toting up to smooth changes.

Mathematical formulation of the problem

The problem can be stated simply as:

$\max_{x \in X} f(x)$ to maximize some variable such as product throughput

or

$\min_{x \in X} f(x)$ to minimize a cost function

where

$f : R^n \to R$

$X \subseteq R^n$.

Methods for solving the problem

Rider the intent function f is linear and the constrained space is a polytope, the problem is a linear programming problem, which possibly will be solved using renowned linear programming way outs.

Stipulating the objective function is concave (maximization problem), or convex (minimization problem) and the constraint set is convex, and then the program is called convex and general methods commencing convex optimization can be worn in most cases.

Proviso the intent function is a ratio of a concave and a convex function (in the maximization case) and the constraints are convex, then the quandary can be transformed to a convex optimization problem by means of fractional programming techniques.

Quite a lot of methods are available for solving non convex problems. One loom is to exploit special formulations of linear programming problems. An extra method involves the exploit of branch and bound techniques, where the program is at odds into subclasses to be solved with convex (minimization problem) or linear approximations that form a lower bound on the on the whole cost within the subdivision. With consequent divisions, at some point an tangible way out will be obtained whose outlay is equal to the best lower bound obtained for any of the approximate way outs. This way out is optimal, although possibly not unique. The algorithm may also be stopped early, with the assurance that the best possible way out is within a lenience from the best point found; such points are called ε-optimal. Terminating to ε-optimal points is on average necessary to make certain finite termination. This is particularly exploitful for large, difficult problems and problems with tentative costs or values where the vagueness can be estimated with an appropriate dependability estimation. Under differentiability and constraint qualifications,the Karush–KuhTucker(KKT) conditions afford necessary conditions for a way out to be optimal. Under convexity, these conditions are also sufficient.

PROBLEM STATEMENT

System and menace Model and thus we consider a computation outsourcing architecture concerning two different entities, as illustrated the cloud customer, who has hefty amount of computationally expensive LP problems to be outsourced to the cloud; the cloud server (CS), which has significant computation resources and provides efficacy computing services, such as hosting the public LP solvers in a pay-per-exploit manner. The customer has a across-the-board linear programming problem Φ (to be formally defined later) to be solved. Moreover, due to the lack of computing assets, like processing power, memory, and storage etc., he may not carry out such pricey computation locally. Hence, the customer resorts to CS for solving the LP computation and leverages its computation 3 capacity in a pay-per-exploit manner. As an alternative of directly sending original problem Φ, the customer first exploits a secret K to map Φ into some encrypted description ΦK and outsources problem ΦK to CS. CS then exploits its public LP solver to get the retort of ΦK and provides a precision proof Γ, but it is made-up to learn nothing or little of the sensitive information contained in the original problem description Φ. Following receiving the way out of encrypted problem ΦK, the customer should be able to first bear out the answer via the appended proof Γ. But it's correct, he then exploits the secret K to plot the output into the desired answer for the original problem Φ.

NLP techniques are exploited consistently and are predominantly efficient in solving optimal direct problems. During the crate of a *discrete* control problem, i.e., when the controls are exerted at discrete points, the problem can be directly stated as a NLP problem. In a *unbroken* control problem, on the supplementary , i.e., when the reins are *functions* to be exerted over a granted planning horizon, an *approximate* way out can be instigate by solving a NLP problem. Throughout this section, we shall consider the following NLP problem:

$$\min_x \quad f(x)$$
$$\text{s.t.} \quad g(x) \geq 0$$
$$h(x) = 0$$
$$x \square X$$

whereX is a subset of IR^{nx}, x is a vector of nx components x1; : : : ; xnx, and f : X → IR,
g : X → IR^{ng} and h : X → IR^{nh} are defined on X.
The function f is generally called the *objective function* or *criterion function*. Each of the constraints $g_i(x) \leq 0$, i = 1,….., n_g, is called an *inequality constraint*, and each of the constraints $h_i(x) = 0$, i = 1,…., n_h, is called an *equality constraint*. reminder also that the set X typically includes lower and upper bounds on the variables; the reason for untying variable limits from the other inequality constraints is that they can play a exploitful role in some algorithms, i.e., they are handled in a detailed way. This vector x □ X satisfying all the constraints is called a *feasible way out* to the problem; and the collection of all such points forms the *feasible region*. The NLP problem, then, is to find a feasible point x* such that $f(x) \geq f(x^*)$ for each feasible point x. Unneeded to say, a NLP problem can be stated as a maximization problem, and the inequality constraints can be on paper in the form $g(x) \geq 0$.

DESIGN GOALS

To facilitate secure and practical outsourcing of LP under the abovementioned model, our method design should achieve the following security and performance guarantees.
1) Correctness: whichever cloud server that loyally follows the method must produce an yield that can be decrypted and verified lucratively by the customer.
2) Soundness: Refusal cloud server can engender an incorrect yield that can be decrypted and verified lucratively by the customer with non-negligible probability.
3) Input/output privacy: No susceptible information from the customer's private data can be resultant by the cloud server during performing the LP computation.
4) Efficiency: The confined computations done by customer should be to a large extent less than solving the original LP on his own. The computation saddle on the cloud server should be within the equivalent time intricacy of existing practical algorithms solving LP problems.

BACKGROUND ON LINEAR AND NON LINEAR PROGRAMMING

An optimization problem is frequently formulated to a mathematical programming problem that seeks the standards for a set of decision variables to minimize (or maximize) an objective function in place of the cost subject to a set of constraints. Intended for linear programming, the objective function is an affine function of the verdict variables, and the constraints are a system of linear equations and inequalities. while a constraint in the outline of a linear inequality can be articulated as a linear equation by introducing a non-negative slack variable, and a free decision variable can be articulated as the difference of two non-negative auxiliary variables, any linear programming problem can be articulated in the following standard form,
minimize $c^T x$ subject to $Ax = b$, $x \geq 0$.
Here x is an n×1 vector of decision variables, A is an m×n matrix, and both c and b are n×1 vectors. It can be tacit further that $m \leq n$ and that A has packed row rank; otherwise, extras rows can always be eliminated from A. In this we study a more universal form as follows,
Minimize $c^T x$ subject to $Ax = b$, $Bx \geq 0$.
In fastidious, a common way out framework based on central-point NLP solvers and sensitivity concepts is well thought-out. Within the subsequent section, we commence some basic concepts and details and portray specific formulations of the MHE and NMPC nonlinear programming problems. During NLP solvers and near some basic NLP sensitivity results. Within Section it derives advanced-step approximation strategies for MHE and NMPC, based on NLP sensitivity to condense on-line computational time. We may also discuss their wide-ranging constancy and performance properties, specially when both are applied together. In Section 5, the potential of the combined MHE and NMPC way out framework is verified on a largescale case study concerning the instantaneous monitoring and control of a distributed low-density polyethylene tubular reactor.
preliminary from an primary point, a direction of pressure group is determined according to a preset rule, and then a move is finished in that direction so that the objective function value is condensed; next to the new point, a new direction is resolute and the process is repeated. The main distinction between these algorithms rest with the rule by which successive directions of movement are selected. A characteristic is usually made stuck between those algorithms which conclude the search direction without using gradient information (*gradient-free* methods), and those using grade (and higher-order derivatives) information (*gradient-based* methods). Here, we shall center of notice our attention on the latter division of methods, and more specifically on Newton-like algorithms

PROPOSED SYSTEM:

The simulated annealing (SA) technique (Kirkpatrick et al., 1983) was derived from statistical mechanics for result near globally-minimum-cost way outs to large optimization problems. It simplifys the hill climbing methods and eliminates their main disadvantage: dependence of the way out on the starting point, and statistically promises to deliver an optimal way out. This is achieved by introducing a probability _ of acceptance (i.e., replacement of the current

point by a new point): _ = 1, if the new point provides a better value of the objective function; moreover, _ > 0, if not. during the latter case, the odds of acceptance _ is a function of the values of objective function for the current point and the new point, and an additional control parameter, \temperature", T. Into general, the lower temperature T is, the smaller the probability for the acceptance of a new point are. During execution of the algorithm, the temperature of the system, T, is lowered in steps.

Mechanism Design Framework

We set forward to pertain problem transformation for mechanism design. The general framework is adopted from a basic approach, whereas our instantiation is completely unusual and novel. In this framework, the progression on cloud server can be represented by algorithm ProofGen and the progression on customer can be ordered into three algorithms (KeyGen, ProbEnc, ResultDec). These four algorithms are summarized lower and will be instantiated later.

• KeyGen(1k) → {K}. *This is a randomized key generation algorithm which takes a system security parameter* k*, and returns a secret key* K *that is exploitd later by customer to encrypt the target LP problem.*

• ProbEnc(K,□) → {□,K}. *This algorithm encrypts the input tuple _ into* _K *with the secret key* K*. According to problem transformation, the encrypted input* _K *has the same form as _, and hence defines the problem to be solved in the cloud.*

• ProofGen(□,K) → {(y, Γ)}. *This algorithm augments a generic solver that solves the problem* □K *to produce both the output* y *and a proof* Γ*. The output* y *later decrypts to* x*, and* Γ *is exploitd later by the customer to verify the correctness of* y *or* x.

• ResultDec(K,□, y, Γ) → {x,⊥}. *This algorithm may choose to verify either* y *or* x *via the proof* Γ*. In any case, a correct output* x *is produced by decrypting* y *using the secret* K*. The algorithm outputs* ⊥ *when the validation fails, indicating the cloud server was not performing the computation faithfully.*

BASIC TECHNIQUES

Convexity and Constraint Qualification

In wide-ranging we can only imagine that an optimization algorithm computes a local minimum and not a global one, i.e. a point $x\_$ with $f(x\_) \le f(x)$ for all $x \in P \cap U(x\_)$, somewhere $U(x\_)$ is a fitting neighborhood of $x\_$. Moreover, each local minimum of a nonlinear programming problem is a global one if the problem is convex, for example, if f is convex, g_j linear for $j = 1, \ldots, me$ and g_j concave for $j = me+1, \ldots, m$. These conditions force the feasible region P to be a convex set.

Definition 1 *A function* $f : Rn \to R$ *is called convex, if*
$f(\lambda x + (1-\lambda)y) \le \lambda f(x) + (1-\lambda) f(y)$
for all $x, y \in Rn$ *and* $\lambda \in (0, 1)$*, and concave, if we replace '≤' by '≥' in the above inequality.* For a twofold differentiable function f convexity is comparable to the the property that $\nabla^2 f(x)$ is positive semi definite, i.e. $z^T \nabla^2 f(x) z \ge 0$ for all $z \in Rn$. Convexity of an optimization problem is imperative mainly from a exploratory point of view, since many convergence, duality or other theorems can be proved only for this special case. Inside practical situations, moreover, we include hardly a chance to analysis whether a numerical problem is convex or not. On the way to be able to formulate the ensuing optimality conditions, we need a special postulation to let alone irregular behavior of the feasible sets P at a local way out. We call it *constriction* 5 *qualification*, to be measured as some kind of *regularity* in more general form. During our situation it is adequate to proceed from the following definition:

Definition 2 *A constraint qualification in* $x\_ \in P$ *is satisfied, if the gradients of active constraints,*
i.e. the vectors $\nabla g_j(x\_)$ *for* $j \in \{1, \ldots, me\} \cup I(x\_)$*, are linearly independent.*

2.2 Karush-Kuhn-Tucker Conditions

Intended for increasing and understanding an optimization method, the ensuing theorems are essential.They characterize optimality and are as a result also important for testing a current iterate with esteem to its convergence accuracy.

Theorem 1 (necessary second order optimality conditions) *Let f and gj be twice continuously differentiable for* $j = 1, \ldots, m$*,* $x\_$ *a local minimizer of (2) and the constraint qualification*
in $x\_$ *be satisfied. Then there exists a* $u\_ \in Rm$*, such that*

a) $u\_$
$j \ge 0$, $j = me+1, \ldots, m$,
$g_j(x\_) = 0$, $j = 1, \ldots, me$,
$g_j(x\_) \ge 0$, $j = me+1, \ldots, m$,
$\nabla_x L(x\_, u\_) = 0$,$u\_$
$j\, g_j(x\_) = 0$, $j = me+1, \ldots, m$
b) $s^T \nabla^2 {}_x L(x\_, u\_) s \ge 0$ (5)
for all $s \in Rn$ *with* $\nabla g_j(x\_)^T s = 0$, $j \in \{1, \ldots, me\} \cup I(x\_)$ *(second order condition).*

Statement a) of the theorem is calledthe *Karush-Kuhn-Tucker-condition*. It says that at a
local way out the incline of the objective function can be articulated by a linear permutation of gradients of active constraints. Moreover statement b) implies that the Lagrangian function is positive semi-definite on the divergent space defined by the active constraints. meant for a discussion of general duality-based optimality conditions see..

Theorem 2 (sufficient second order optimality conditions) *Let f and gj be twice continuously differentiable for* $j = 1, \ldots, m$ *and* $x\_ \in Rn$, $u\_ \in Rm$ *be given, so that the following conditions are satisfied:*
a) $u\_j \ge 0$, $j = me+1, \ldots, m$,
$g_j(x\_) = 0$, $j = 1, \ldots, me$,
$g_j(x\_) \ge 0$, $j = me+1, \ldots, m$,
$\nabla_x L(x\_, u\_) = 0$,
$u\_j\, g_j(x\_) = 0$, $j = me+1, \ldots, m$ *(first order condition),*
b) $s^T \nabla^2 {}_x L(x\_, u\_) s > 0$ *for all* $s \in Rn$ *with* $s\_ = 0$, $\nabla g_j(x\_)^T s = 0$, $j = 1, \ldots, me$*, and for all s with* $\nabla g_j(x\_)^T s = 0$, $j = me+1, \ldots, m$*, and* $u\_j > 0$ *(second order condition).*

Then x_ is an isolated local minimum of f on P, i.e. there is a neighborhood U(x_) of x_ with f(x_) < f(x) for all x ∈ U(x_) ∩ P, x _= x_.

while reading a nonlinear programming textbook, one has to be attentive of the fact that the optimality conditions are often stated in a vaguely different way. The formulation of a NLP problem varies from author to author depending e.g. whether a minimum or a maximum is search, whether the inequality constraints exploit ≤ insteadof ≥, or whether upper and lo wer bounds are included. moreover there exist different version of the above statements, where the assumptions are whichever more general or more expert, respectively.

CONCLUDING REMARKS

In this paper, for the first time, extend our result to non-linear programming computation outsourcing in cloud. we formalize the problem of securely outsourcing and solving NLP computations in cloud computing, and provide such a practical mechanism design which fulfills

input/output privacy, cheating resilience, and efficiency. Many NLP problem solving and optimization techniques have been proposed and explained.

The problem solving methods demonstrates the immediate practicality of the proposed mechanism. We plan to investigate some interesting future work as

follows: 1) devise robust algorithms to achieve numerical stability; 2) explore the sparsity structure of problem for further efficiency improvement; 3) establish formal security framework;

REFERENCES

[1] P. Mell and T. Grance, "Draft nist working definition of cloud computing,"Referenced on Jan. 23rd, 2010 Online at http://csrc.nist.gov/groups/SNS/cloud-computing/index.html, 2010.

[2] Cloud Security Alliance, "Security guidance for critical areas of focus in cloud computing," 2009, online at http://www.cloudsecurityalliance.org.

[3] C. Gentry, "Computing arbitrary functions of encrypted data," *Commun.ACM*, vol. 53, no. 3, pp. 97–105, 2010.

[4] Sun Microsystems, Inc., "Building customer trust in cloud computing with transparent security," 2009, online at https://www.sun.com/offers/ details/sun transparency.xml.

[5] M. J. Atallah, K. N. Pantazopoulos, J. R. Rice, and E. H. Spafford, "Secure outsourcing of scientific computations," *Advances in Computers*, vol. 54, pp. 216–272, 2001.

[6] S. Hohenberger and A. Lysyanskaya, "How to securely outsource cryptographic computations," in *Proc. of TCC*, 2005, pp. 264–282.

[7] M. J. Atallah and J. Li, "Secure outsourcing of sequence comparisons," *Int. J. Inf. Sec.*, vol. 4, no. 4, pp. 277–287, 2005.

[8] D. Benjamin and M. J. Atallah, "Private and cheating-free outsourcing of algebraic computations," in *Proc. of 6th Conf. on Privacy, Security,and Trust (PST)*, 2008, pp. 240–245.

[9] R. Gennaro, C. Gentry, and B. Parno, "Non-interactive verifiable computing:Outsourcing computation to untrusted workers," in *Proc. Of CRYPTO'10*, Aug. 2010.

[10] M. Atallah and K. Frikken, "Securely outsourcing linear algebra computations," in *Proc. of ASIACCS*, 2010, pp. 48–59.

[11] A. C.-C. Yao, "Protocols for secure computations (extended abstract)," in *Proc. of FOCS'82*, 1982, pp. 160–164.

[12] Craig Gentry, "Craig's Thesis - A FULLY HOMOMORPHIC ENCRYPTION SCHEME", 2009

[13] D. Luenberger and Y. Ye, *Linear and Nonlinear Programming*, 3rd ed. Springer, 2008.

[14] C. Wang, N. Cao, J. Li, K. Ren, and W. Lou, "Secure ranked keyword search over encrypted cloud data," in *Proc. of ICDCS'10*, 2010.

[15] S. Yu, C. Wang, K. Ren, and W. Lou, "Achieving secure, scalable, and fine-grained access control in cloud computing," in *Proc. of IEEE INFOCOM'10*, San Diego, CA, USA, March 2010.

[16] S. Boyd and L. Vandenberghe, *Convex Optimization*. Cambridge University Press, 2004.

[17] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein, *Introduction to Algorithms*, 2nd ed. MIT press, 2008.

[18] V. Strassen, "Gaussian elimination is not optimal," *Numer. Math.*, vol. 13, pp. 354–356, 1969.

[19] D. Coppersmith and S. Winograd, "Matrix multiplication via arithmetic progressions," in *Proc. of STOC'87*, 1987, pp. 1–6.

[20] MOSEK ApS, "The MOSEK Optimization Software," Online at http: //www.mosek.com/, 2010.

[21] P. Paillier, "Public-key cryptosystems based on composite degree residuosity classes," in *Proc. of EUROCRYPT'99*, 1999, pp. 223–238.

[22] S. Even, O. Goldreich, and A. Lempel, "A randomized protocol for signing contracts," *Commun. ACM*, vol. 28, no. 6, pp. 637–647, 1985.

[23] W. Du and M. J. Atallah, "Secure multi-party computation problems and their applications: a review and open problems," in *Proc. of New Security Paradigms Workshop (NSPW)*, 2001, pp. 13–22.

[24] J. Li and M. J. Atallah, "Secure and private collaborative linear programming," in *Proc. of CollaborateCom*, Nov. 2006.

[25] P. Golle and I. Mironov, "Uncheatable distributed computations," in *Proc. of CT-RSA*, 2001, pp. 425–440.

[26] "Mixed-Integer Nonlinear Programming", Michael R. Bussieck Armin Pruessner_February 19, 2003