Decomposing the optimization of a gas lifted oil well network

Andreas Grothey, Ken McKinnon

March 2000

MS-00-005

Supported by EPSRC research grant GR/K51587

Department of Mathematics and Statistics
University of Edinburgh, The King's Buildings, Edinburgh EH9 3JZ
Tel. (44) 131 650 5747 or 5042 E-Mail : agr or ken@maths.ed.ac.uk

Decomposing the optimization of a gas lifted oil well network

A. Grothey, K.I.M McKinnon

March 2000

Abstract

Low pressure oil wells can be made to flow by the injection of gas into the well. The resulting flow from the well increases discontinuously with the amount of gas injected. A common way to model this is by introducing integer variables which results in a MINLP formulation. Such a problem can be solved efficiently by a specialised Branch & Bound code. However for more than about 200 wells a complete search of the Branch & Bound tree becomes intractable. In this report we show that problems with a certain structure can be solved by different decomposition methods, one of which uses a bundle method to solve the resulting master problem. Improvements of the solution times up to a factor of 100 were observed on the largest problems which could be solved without decomposition, and problems with 1000 wells were solved by the decomposition method. We also show that there is substantial scope for the parallelization of such a method.

1 Introduction

Often oil wells do not flow naturally. This occurs increasingly as an oil field ages and its pressure declines. A common way of making such wells flow is to inject gas (at high pressure) deep into the well and the resulting reduction in the density of the oil causes the well to flow. Typically however small amounts of injected gas produce no output from the well, but once a threshold "kick off" value is reached the well output jumps to a positive level in a discontinuous manner. As the injected gas increases further the output increases smoothly to a maximum. Oil exploration systems consist of complex networks of such wells often with over 1000 wells in the network. Apart from oil the wells also produce different quantities of gas and water. These components are gathered and separated in a separation unit which usually has a limited capacity. Some of the gas will be used to power a compressor to compress the remaining gas to the high pressure required for the injection. Alternatively a limited amount of the produced gas can be sold to the market at a given price. The aim of this report is to describe methods of solving such a network for an optimal operating strategy.

The network could be modelled as a nonlinear programming problem (NLP) apart from the additional difficulty introduced by the discontinuous behaviour of the gas-lifted oil wells. A possible way to overcome this problem is by the introduction of integer variables and subsequent reformulation as a Mixed Integer Nonlinear Program (MINLP). Such a model can be solved by a Branch & Bound algorithm but even for a specialised code the problem becomes intractable for more than about 200 wells. In this report we suggest two decomposition approaches related to Benders' Decomposition whose subproblems are still MINLPs but of a smaller size. The master problems are solved either by a superlinearly convergent bundle method or by resolving the whole problem for fixed values of the integer variables. With these approaches solution time on 300 wells could be reduced by about a factor of 100 and we were able to solve problems of up to 1000 wells which could not be solved without decomposition. While there is some possibility that our approach only finds a local solution to the problem, we also give a Lagrangian Relaxation of the problem which proved that our solutions are not further than 0.02% from the optimal in the worst case.

In the next section we present the full modelling of the oil exploration network as an NLP problem with a discontinuous constraint. A reformulation as a MINLP will be given as well. Section 3 discusses the suggested decomposition approach in detail, while section 4 summarises the algorithms. In the final section computational results of the method in comparison to a specialised Branch & Bound code are given.

2 Problem formulation

Let the total outflow from well i be denoted by

$$q_i^{out} = F_i(q_i^{in}; p_i^{in}, p_i^{out}) \tag{1}$$

which is a function of the injected gas q_i^{in}, and the pressures p_i^{in}, p_i^{out} in the pipes leading to and away from the well. As mentioned F_i is a discontinuous function in

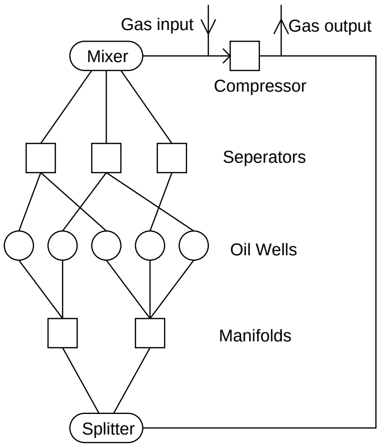

Figure 1: A sub-network of an oil and gas gathering network

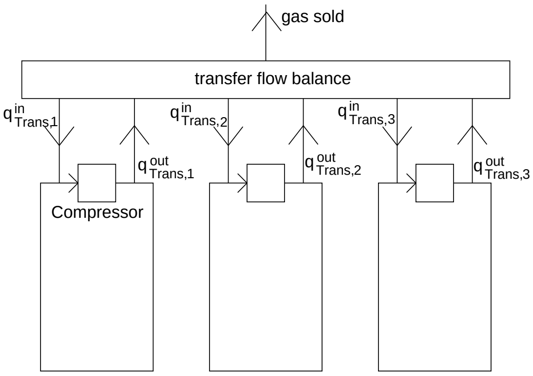

Figure 2: Three sub-networks linked by an interconnector

q_i^{in}, being zero for small values of q_i^{in} up to a threshold value $\overline{q}_i$ at which F_i jumps to a positive level, as illustrated in figure 3. The total flow F_i consists of fractions r_i^g, r_i^o, r_i^w of gas, oil and water with $r_i^g + r_i^o + r_i^w = 1$.

In an oil gathering network (as seen in figures 1,2) injection gas is supplied to the wells via a number of manifolds, each supplying gas to different wells. The output from the wells is led to several separation units which have a limited capacity to process liquid. This limit is

$$\sum_{i \in I_k} (r_i^w + r_i^o) F_i \leq C_k, k \in \text{Separators}, \tag{2}$$

where I_k is the set of wells supplying gas to separator k. The gas fractions are

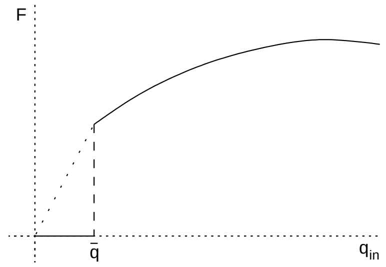

Figure 3: Oil well flow as function of injected gas

gathered in a mixing unit and passed on to the compressor. The compressor burns some of the gas to increase the pressure of the remaining gas according to

$$\kappa_g q_{burn} = q_C \frac{\ln(p_C^{in}) - \ln(p_C^{out})}{p_C^{in} - p_C^{out}} \tag{3}$$

where κ_g is a constant which depends on the specific energy of the gas and the efficiency of the compressor. The compressed gas is then used as injection gas into the wells after passing through a splitting unit and the manifolds. Further there are pressure drop constraints

$$(p_j^{in})^2 = (p_j^{out})^2 + q_j^2/\theta_j \tag{4}$$

along all pipes j connecting separators, mixer, compressor, splitter and manifolds, where θ_j is a pipe specific constant depending on its length and diameter. At all units l there are also flow balance constraints

$$\sum_{j \in I_l^{in}} q_j = \sum_{j \in I_l^{out}} q_j \tag{5}$$

where I_l^{in}, I_l^{out} are the sets of pipes j passing in and out respectively of unit l.

There may be several of these networks which are linked via an interconnecting unit as shown in figure 2. This takes gas from the high-pressure end of the compressor in gas rich networks and feeds it into the low-pressure end of networks which are short of injection gas. Gas which is not needed can also be sold to the market, although there is an upper limit $\overline{q}_{sell}$ on the amount of gas that can be sold. In the model the interconnector is simply modelled by another flow balance constraint.

Altogether the model is as follows:

$$\max c_g q_{sell} + c_o \sum_{\substack{t\in\text{Networks}\\ i\in\text{Wells}}} r_i^o F_{i,t} \tag{6a}$$

subject to

$$\sum_{j\in I_l^{in}} q_j = \sum_{j\in I_l^{out}} q_j, \qquad \forall l \in \text{Units} \tag{6b}$$

$$(p_j^{in})^2 = (p_j^{out})^2 + q_j^2/\theta_j, \qquad \forall j \in \text{Pipes} \tag{6c}$$

$$\kappa_g q_{burn,t} = q_{C,t}^{in} \frac{\ln(p_{C,t}^{in}) - \ln(p_{C,t}^{out})}{p_{C,t}^{in} - p_{C,t}^{out}}, \qquad \forall t \in \text{Networks} \tag{6d}$$

$$C_k \geq \sum_{i\in I_k} (r_i^w + r_i^o) F_i, \qquad \forall k \in \text{Separators} \tag{6e}$$

$$q_i^{out} = r_i^g F_i(q_i^{in}; p_i^{in}, p_i^{out}), \qquad \forall i \in \text{Wells} \tag{6f}$$

$$q_{sell} \leq \overline{q}_{sell} \tag{6g}$$

where c_g, c_o are the market prices for gas and oil respectively. Problem (6) is a nonlinear nonconvex programming problem, incorporating the discontinuous constraint (6f).

2.1 MINLP model

In this formulation (6f) is still a discontinuous constraint, so a standard NLP method may fail to converge or get trapped in local optima. A way around this is to reformulate the problem as an MINLP by rewriting constraint (6f) as

$$\left.\begin{aligned} q_i^{out} &= r_i^g \lambda_i F_i(\hat{q}_i^{in}; p_i^{in}, p_i^{out}) \\ q_i^{in} &= \lambda_i \hat{q}_i^{in} \\ \hat{q}_i^{in} &\geq \overline{q}_i, \quad \lambda_i \in \{0,1\}. \end{aligned}\right\} \qquad \forall i \in \text{Wells} \tag{6f$'$}$$

In what follows the resulting modification of (6) where constraint (6f) is replaced with (6f′) is referred to as (6′). This model can now be solved by a Branch & Bound code as described in the following section. We just mention here that by relaxing the integrality constraint on λ_i to $\lambda_i \in [0,1]$ we obtain a smooth NLP relaxation of the problem which can be solved by an NLP solver to derive upper bounds on the value of the problem. This relaxation corresponds to substituting the well outflow function F_i with an outer approximation, which is shown as a dashed line in figure 3. Although the new model ($6'$) is nonconvex, we assume that for a fixed choice of integer variables (or the integrality constraints relaxed) there are no local solutions, so that a local method such as SQP will find the global solutions for all nodes in the Branch & Bound tree. In our experiments with the problem we have indeed never found a local solution.

3 Methods

3.1 Branch & Bound

We have applied a Branch & Bound algorithm to solve problem ($6'$). This method used pseudo costs of moving integer variables to integrality and an estimate of their variance to find an efficient branching strategy. It was found to be very effective at finding good solutions early in the search. However its applicability is limited to problems of about 200 wells or less: e.g. a 200 well problem took 218 minutes and 1053 nodes to solve. Bigger problems become quickly intractable, a trial run on 300 wells crashed in the nonlinear solver after 48h having examined 7450 nodes, although the optimal node had already been found (but not proven to be optimal).

It is important to be aware that the main difficulty in solving this problem is due to the integrality constraint on the λ_i in ($6f'$). If this is relaxed the problem becomes much easier and possible to solve by an NLP code. However even in this case decomposition might still be beneficial as will be described later.

3.2 Decomposition

To overcome the rapid increase of solution time with the size of the problem which is observed for Branch & Bound, a decomposition strategy has been devised.

In problem ($6'$) the sub-networks would be independent of each other if it were not for the link through the interconnector. Two obvious strategies to decompose the problem are therefore to temporarily fix the gas transfers between networks (Benders' type decomposition) or to relax the mass balance constraint in the interconnector (Lagrangian Relaxation). Both methods will be studied in this section.

3.2.1 Benders' Decomposition

Fixing the transfer flows $q_{Trans,t}^{in/out}$ in figure 2 between the sub-network decomposes the problem into subproblems (8) modelling its independent parts and a master problem (7) which controls the transfer flows:

$$\begin{aligned} \max \quad v(q) = c_g q_{sell} + \sum_{t \in \text{Networks}} v_t(q_{Trans,t}^{in}, q_{Trans,t}^{out}) \\ \text{subject to } q_{sell} + \sum_t q_{Trans,t}^{in} = \sum_t q_{Trans,t}^{out} \\ q_{sell} \leq \overline{q}_{sell} \end{aligned} \tag{7}$$

where

$$\begin{aligned} v_t(q_{Trans,t}^{in}, q_{Trans,t}^{out}) = \max \quad & c_o \sum_{i \in \text{Wells}} r_{i,t}^o F_{i,t} \\ \text{subject to} \quad & \\ & \text{constraints (6b - 6e/6f}') \text{ for network } t. \end{aligned} \tag{8}$$

The subproblems in this scheme are still MINLPs but of a much smaller size than the original problem.

The subproblem value functions v_t are in general not continuous. This is because the optimal choice of integer variables λ for some value of q_{Trans} might become infeasible as q_{Trans} changes, thus resulting in the next best choice of integer variables becoming optimal with a possible discontinuity in the value function. However the value function is guaranteed to be continuous if we can ensure that there are feasible points for all values of q_{Trans} and λ (see [5] for a proof). We could ensure this by allowing all subproblems to buy their gas shortfall from the market (at a high cost). In this case v_t will also be piecewise differentiable. In our experiments we never observed discontinuous value functions and therefore used the unmodified model throughout.

Assuming continuous subproblem value functions, the master problem (7) is a problem of maximizing a nonsmooth nonconcave function subject to a linear constraint. We have used two different methods to solve this master problem in the decomposition, both of which are described below.

3.2.2 Master problem solved by a Bundle Method (BM)

The standard methods for solving nonsmooth optimization problems are bundle methods developed by Kiwiel, Lemarechal and Zowe [3, 4, 6]. Their idea is to keep a *bundle* of subgradients and function values $\{(g_j, f_j)\}, g_j \in \partial v(q_j), f_j = v(q_j)$ at past points $\{q_j\}$. Where

$$\partial v(q) := \text{conv}\{g : \exists g_i \to g, g_i = \nabla v(q_i), q_i \to q\}$$

is the subgradient of v at q. These can be used to define a cutting plane approximation of the function $v(q)$ based at the current point q_k

$$f_{cp}^{(k)}(q) = \min_j \{f_j + g_j^T(q - q_j) - \alpha_{k,j}\} \tag{9}$$

$$\alpha_{k,j} = \min\{|f_k - f_j - g_j^T(q^{(k)} - q^{(j)})|, \gamma\|q^{(j)} - q^{(k)}\|^2\} \tag{10}$$

where the $\alpha_{j,k}$ are *subgradient locality measures* to allow for the nonconcave nature of $v(q)$. Bundle methods use $f_{cp}^{(k)}$ plus a regularizing term $1/2d^T d$ to solve for the new step

$$\begin{aligned} \max_d \quad & f_{cp}^{(k)}(q^{(k)} + d) + 1/2d^T d \\ & \text{subject to } \|d\| \leq \rho \end{aligned} \tag{11}$$

If the new point $q_{k+1} = q_k + d^*$ achieves a better objective value than q_k the subgradient at q_{k+1} will be added to the bundle (SERIOUS STEP). Otherwise the algorithm can decide whether to include it in the bundle anyhow to improve the model (NULL STEP), or to just resolve the model for a smaller trust region radius ρ. Note that (11) can be reformulated as a QP and therefore solved efficiently.

Bundle methods will typically only converge linearly. However by using second derivatives $H_j \in \partial^2 v(q_j)$, where

$$\partial^2 v(q) := \text{conv}\{H : \exists H_i \to H, H_i = \nabla^2 v(q_i), q_i \to q\}$$

is the *sub-Hessian* of v at q analogously defined to the subgradient, superlinear convergence can be obtained. The master problem (11) becomes in this case:

$$\begin{aligned} \max_d \quad & f_{sup}^{(k)}(q^{(k)} + d) + 1/2d^T \left[\sum_j \lambda_j H_j\right] d \\ & \text{subject to } \|d\| \leq \rho \end{aligned} \tag{12}$$

where

$$f_{sup}^{(k)}(q) = \max_j \left\{f_j + g_j(q - q^{(j)}) + [g_j + H_j(q^{(k)} - q^{(j)})]^T(q - q^{(k)}) - \alpha_{j,k}\right\} \tag{13}$$

Details of this method are described in an earlier report [2].

3.2.3 Master problem solved by Transfer Resolving (TR)

Bundle methods only use a first (or second) order approximation of the functions $v_t(q)$ in the master problem (7) at the current point to obtain a new set of transfers q_{Trans}. In our case, however, it is possible to use a better approximation of the subproblem value functions $v_t(q)$. To this end denote by $\hat{v}_t^{(k)}(q)$ the value of the sub-network t if the integer variables λ_i are fixed at their optimal values in the k-th sub-network solve

$$\begin{aligned}
\hat{v}_t^{(k)}(q) = \max c_o & \sum_{i \in Wells} r^o_{i,t} F_{i,t} \\
& \text{subject to} \\
& \qquad \text{constraints (6b - 6e/6f}') \text{ for network } t \\
& \qquad \lambda_i = \lambda_i^{*(k)}.
\end{aligned} \tag{14}$$

The function $\hat{v}_t^{(k)}(q)$ is identical to $v_t(q)$ as long as $\lambda_i^{*(k)}$ are still optimal for q and is an upper bound for $v_t(q)$ for all values of q. Using $\hat{v}_t^{(k)}(q)$ as our approximation of the subproblem value function in the master problem we obtain

$$\begin{aligned}
\max_q \quad & c_g q_{sell} + \sum_t \hat{v}_t^{(k)}(q_{Trans}) \\
& \text{subject to } q_{sell} \leq \overline{q}_{sell}
\end{aligned} \tag{15}$$

This problem could be solved by a decomposition method employing a bundle method as described in the previous section. However, since it is a purely continuous NLP it is possible to solve it as a whole undecomposed problem by an SQP method. This can be done efficiently by hotstarting from the last subproblem solutions.

Note that since the subproblem solutions from (8) are feasible in (15, 14), their solution has to improve the best known value. Conversely the solution of (15, 14) is feasible in the subproblems (8), so resolving them will also improve the best known value. Therefore the process is guaranteed to converge. This scheme can be thought of as a one-at-a-time optimization on the whole problem (6), keeping alternating transfer flows q_{Trans} or integer variables λ_i fixed and optimising for the other set of variables.

In both methods starting values for the transfer flows can be obtained by solving (6) with the integrality constraints dropped. This can be done either as a single problem or decomposed in the same way as (15, 14). For large problems solving this undecomposed NLP becomes the dominant cost in the algorithm due to the fact that at the beginning no subproblem solutions for the hotstart are available. This can be overcome by first solving all subproblems for zero values of the transfer flows and using these to hotstart the first undecomposed NLP.

As problem sizes increase, even when using hotstarts the NLPs for the entire network become the most expensive part of the algorithm. In this case a switch to solving these as a decomposed problem, or a hybrid method where several iterations of a bundle method are performed before the best guess from it is used as a hotstart for the undecomposed NLP, might well be beneficial. This would especially be so in a parallel implementation.

3.3 Lagrangian Relaxation (LR) to obtain upper bounds

Since the decomposition approaches are related to a one-at-a-time optimization they are not guaranteed to find the global solution. This is the case even though we assume that both the subproblem and the master in the decomposition can be solved to global optimality. This is illustrated by figure (4): For a one-at-a-time

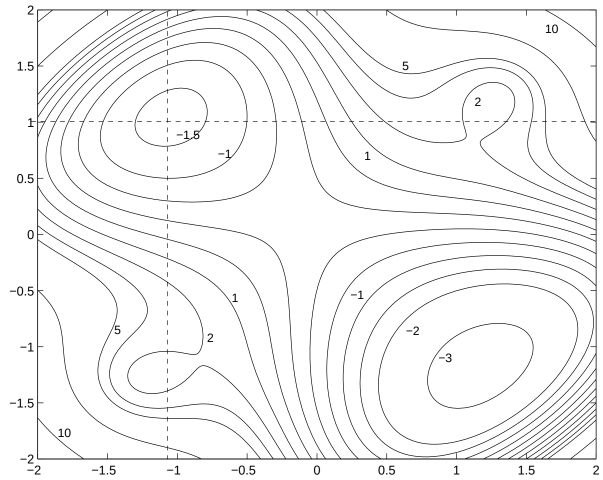

Figure 4: one-at-a-time method fails to find global solution

method the local solution in the top left hand corner is globally optimal in both search direction, but the method will fail to find the true global solution in the bottom right hand corner.

There are several different ways to obtain upper bounds on the optimal objective function value which can be used to asses the quality of the best found solution. An obvious way is to solve (6) with the integrality constraints dropped. However much better bounds were obtained by a Lagrangian Relaxation of the problem. This is obtained by dualising the transfer flow-balance constraint and leads to the problem

$$\begin{aligned} \max \quad & c_g q_{sell} + \sum_t c_o r_i^o F_i + \mu \left(\sum_t q_{Trans,t}^{in} + q_{sell} - \sum_t q_{Trans,t}^{out} \right) \\ & \text{subject to constraints (6b - 6e/6f}'\text{)} \\ & \qquad\qquad q_{sell} \leq \overline{q}_{sell}, \end{aligned} \tag{16}$$

which decomposes into

$$\begin{aligned} \max_q \sum_t v_t^{lag}(q) + (c_g + \mu) q_{sell} \\ \text{subject to } q_{sell} \leq \overline{q}_{sell} \end{aligned} \tag{17a}$$

where

$$\begin{aligned} v_t^{lag}(q) = \max \quad & \sum_i c_o r_{i,t}^o F_{i,t} + \mu \left(q_{Trans,t}^{in} - q_{Trans,t}^{out} \right) \\ & \text{subject to constraints (6b - 6e/6f}'\text{) for network } t. \end{aligned} \tag{17b}$$

Duality theory guarantees that the optimal value of (16), or equivalently of (17), is an upper bound on the optimal value of (6) for all values of μ. If there is no duality gap, the tightest bound is obtained for μ equal to the optimal multiplier on the transfer flow-balance constraint in (6). However due to the presence of the duality gap, even for the optimal choice of μ there might be a difference between the bounds derived by (16,17) and the solutions obtained by the decomposition methods, even if these solutions are indeed global.

4 Description of the Algorithm

To summarise the two suggested decomposition methods can be described by the following steps:

1) Solve problem (6) with the integrality constraints relaxed (either decomposed or undecomposed) to obtain initial estimates of the transfer flows q_{Trans}.

2) Solve all subproblems for current estimates of transfer flows.

3) **a) for Bundle-based Decomposition (BD):**
Decide whether to add cuts from the last subproblem call to the current bundle. Solve the Bundle Method master problem (12) for new transfer flows.

b) for Transfer Resolve (TR):
Fix integer variables in (6) to their optimal values in the last subproblem call and resolve (6) for new transfer flows

4) If master problem predicts improvement on subproblem solution go to step 2).

5) convergence

5 Computational Results

We have tested our decomposition approaches on a collection of randomly generated problems ranging from 100 to 1000 wells. In these the well characteristics and sub-network structures are different across the sub-networks. We used the FilterSQP code developped by Fletcher and Leyffer [1] as an NLP solver at steps 1),2),3b) of the algorithm. All testing was done on a 400MHz UltraSparc-II CPU and solution times below are given for this processor.

Problem	no decomposition		BD			TR		
2x50	21	76	4	6.2	5.9	2	3.6	3.3
2x100	218	1053	4	65.4	64.3	2	33.8	32.7
3x100a	>2880(F)	>7450(F)	4	97.3	96.2	2	48.4	47.3
3x100b	-	-	4	271.7	270.3	2	132.9	131.5
4x100	-	-	4	135.2	133.3	2	69.8	67.9
5x100a	-	-	4	159.1	153.1	2	96.0	90.0
5x100b	-	-	7	264.4	258.4	2	96.5	90.5
10x100	-	-	5	564.2	457.9	2	298.8	192.4

Table 1: Solution times (min) for the different methods (F: failure, -: missing results)

Table 1 compares the solution times for the two decomposition methods to the original undecomposed MINLP formulation. For the undecomposed method the columns give solution time and number of nodes examined by the Branch & Bound method. For each decomposition approaches the columns refer to the number of master iterations and solution times for two variants which differ only in how the first relaxed problems is solved. In the first variant the relaxed problem at step 1) is solved from scratch, and in the second variant all the relaxed subproblems are first solved with zero gas transfer and these solutions are then used as a hotstart for for the relaxed problem at step 1), as described in section 3.2.3. All timings are given in minutes.

Clearly the gains from applying decomposition in these examples are enormous. The TR-decomposition reduced the solution time for problem `3x100a` by a factor of 60. This is even a conservative estimate considering that the undecomposed solve failed about halfway through the run, so that a factor of 100 is probably more realistic. No further comparisons can be made with the undecomposed model, due to the drastic increase of solution time with the size of the problem.

On the other hand both decomposition methods could solve the `10x100` well network without problems and for the variant employing hotstarts of the problem at step 1) solution time increases only linearly with the number of sub-networks, so that it should be possible to solve even bigger problems.

The hotstarts of the first problem are only an issue for the biggest problem, but there the difference is significant. For even bigger problems it would be worth considering solving both the problem at step 1) as well as the Master problem at step 3) in (TR) by decomposition using a bundle method.

To show the quality of the solutions found by decomposition, table 2 gives the objective value of the solutions found by the different methods together with the gap (in percent) between the solutions found and the upper bound derived by Lagrangian Relaxation for the optimal value of μ. The gaps between the upper bound and the solution found by the decomposition methods are insignificant. As pointed out earlier we might, and quite likely will, have indeed found the global solution in all cases, with the observed gap being the duality gap due to nonconvexity.

5.1 Towards a parallel implementation

We have made some preliminary experiments with parallelising our methods. The subproblem solves at steps 1),2) and 3) have been distributed among processors and the master problem solves at steps 1) and 3) are solved on all processors simultaneously. So far we have only simulated a parallel implementation on a single processor, but since the communication overhead is minimal (a vector of

Problem	LR	no decomposition		BD		TR	
2x50	209.046	209.046	0.0%	209.046	0.0%	209.046	0.0%
2x100	191.098	191.098	0.0%	191.098	0.0%	191.098	0.0%
3x100a	308.491	(308.491)	(0.0%)	308.491	0.0%	308.491	0.0%
3x100b	304.189	-	-	304.189	0.0%	304.189	0.0%
4x100	415.622	-	-	415.546	0.018%	415.546	0.018%
5x100a	514.812	-	-	514.736	0.015%	514.736	0.015%
5x100b	507.253	-	-	507.216	0.007%	507.216	0.007%
10x100	891.726	-	-	891.614	0.012%	891.614	0.012%

Table 2: Objective values obtained and gap to upper bound

size 100 to be broadcasted for every subproblem solve of approx 5min) we expect to be able to confirm these results in a truly parallel implementation.

no. processors	BD		TR	
1	457.91	1	192.4	1
2	237.40	1.93	108.6	1.77
3	163.85	2.79	79.9	2.41
4	128.02	3.58	67.1	2.87
5	119.01	3.85	59.1	3.25

Table 3: Solution time and potential speed-up of a parallel implementation

Table 3 gives solution times (first column) and speed-ups (second column) on 1-5 processors for both methods on problem `10x100` and shows that there is substantial scope for a parallel implementation of our methods. The bottle-neck in parallelising the TR method is the Master Problem (at step 3) which accounts for almost 10% of the solution time for problem `10x100`. For the BD method about 2% of the solution time is spent in the non-parallelisable full-problem solve at step 1). A future version which solves the relaxation at step 1) and the TR-Master problems by a bundle method will hopefully remedy this.

6 Conclusions

We have presented two approaches of decomposing the MINLP formulation of a network of gas-lifted oil wells. Significant decrease in solution time could be demonstrated using these approaches. Also much bigger problems can be solved using our approaches than could be earlier.

We expect the presented methods to work well up to a size of about 2000 wells. Above that limit even hotstarted NLPs will become too expensive to solve so that a completely decomposed approach using bundle methods to decompose the NLPs would probably become advantageous.

Further work is required on a truly parallel implementation of the methods. Also, we expect to achieve better speed-up for a completely decomposed variant of our method.

References

[1] R. Fletcher and S. Leyffer. Nonlinear programming without a penalty function. Numerical Analysis Report NA/171, Dundee University, 1997.

[2] A. Grothey and K. I. M. McKinnon. A superlinearly convergent trust region bundle method. Technical Report MS98-015, University of Edinburgh, 1998.

[3] K. C. Kiwiel. *Methods of Descent for Nondifferentiable Optimization*. Number 1133 in Lecture Notes in Mathematics. Springer-Verlag, first edition, 1985.

[4] C. Lemaréchal. *Extensions diverses des méthodes de gradient et application*. PhD thesis, Paris, 1980.

[5] R. Meyer. The validity of a family of optimization methods. *SIAM Journal on Control*, 8:41–54, 1970.

[6] J. Zowe and H. Schramm. A version of the bundle idea for minimizing a nonsmooth function: Conceptual idea, convergence analysis, numerical results. *SIAM Journal on Optimization*, 2(1):121–152, 1991.