

Smooth-and-Dive Accelerator: A Pre-MILP Primal Heuristic applied to Scheduling

Jeffrey Dean Kelly¹

February 2002

¹ Honeywell Hi-Spec Solutions, 300 Yorkland Blvd., Toronto, Ontario, Canada, M2J 1S1
E-mail: jeff.kelly@honeywell.com

Abstract

This article describes an effective and simple primal heuristic to greedily encourage a reduction in the number of binary or 0-1 logic variables before an implicit enumerative-type search heuristic is deployed to find integer-feasible solutions to “hard” production scheduling problems. The basis of the technique is to employ well-known smoothing functions used to solve complementarity problems to the local optimization problem of minimizing the weighted sum over all binary variables the product of themselves multiplied by their complement. The basic algorithm of the “smooth-and-dive accelerator” (SDA) is to solve successive linear programming (LP) relaxations with the smoothing functions added to the existing problem’s objective function and to use, if required, a sequence of binary variable fixings known as “diving”. If the smoothing function term is not driven to zero as part of the recursion then a branch-and-bound or branch-and-cut search heuristic is called to close the procedure finding at least integer-feasible primal infeasible solutions. The heuristic’s effectiveness is illustrated by its application to an oil-refinery’s crude-oil blendshop scheduling problem, which has commonality to many other production scheduling problems in the continuous and semi-continuous (CSC) process domains.

Introduction

Providing good production schedules that can be implemented at a plant to decrease cycle-times and cycle-stocks or to increase productivity and predictability, just to name a few, are all benefit areas that manufacturers are continuously looking to improve. Better scheduling techniques and specifically “scheduling optimization” is the major focus of many manufacturers and application vendors. For most competitive markets, simply being feasible or able to produce a product is not good enough. Today manufacturers need to be faster, better and cheaper than their competition and this requires an enormous number of hypothetical schedules to be computed from which the best found schedules are presented to the scheduler. Optimization in the context of scheduling (or planning) implies some sort of automated search applied to a surrogate of the plant’s operation for some finite time horizon in the future. This surrogate is obviously the production scheduling model which in our case is formulated as a discrete-time, mixed 0-1 linear programming problem (MILP). The major difficulty with scheduling optimization is the mixture of both continuous (flows and inventories) and discrete (modes and moves) decision variables that must be searched over, respecting key material balances, logic inequalities and finite-capacity constraints, in order to generate *optimized* or *approximate* schedules. The focus of the smooth-and-dive accelerator (SDA) is simply to decrease the time required to find locally optimized solutions using branch-and-bound or branch-and-cut . Current off-the-shelf MILP codes are very powerful and effective but for very large-scale and highly resource-constrained problems may take an impractical amount of time just to find the first integer-feasible solution even when the problem is well formulated. However, because the SDA is essentially a primal heuristic, meaning that it tries to generate greedily, integer-feasible solutions from fractional solutions of the linear program relaxation, it can terminate as being infeasible which is no surprise given that even finding integer-feasible solutions is known to be *NP-complete* for our

production scheduling problems (Nemhauser and Wolsey, 1988). In addition, because there is no explicit local improving search mechanism besides the post implicit enumerative search, finding better solutions from the SDA incumbent integer-feasible solution is only partly addressed. Hence, the SDA is clearly a locally optimal search paradigm which serves pragmatically to aid in the overall search process for integer-feasible solutions. For most industrial or practical settings, global feasibility of the scheduling problem over some near-term horizon is the name of the game which without an automated search is virtually impossible to achieve using manually generated schedules derived from rules-of-thumb. Although seemingly feasible schedules are being generated by-hand with the aid of spreadsheets for many industrial processes, there are hidden buffers such as time, inventory and capacity as well as complexity reductions such as reducing the possibility for parallel processing and pre-assigning or dedicating equipment to particular tasks when they are really multipurpose which may result in a very inefficient configuration. All of these have evolved or have been designed explicitly into the process in order to help the scheduler meet the demands of the marketplace and the sophistication of the plant. Using the power of MILP for scheduling optimization to reset some of these legacy production inefficiencies is the primary driving force for the enhancement of MILP technology using SDA.

If we represent the production scheduling optimization problem for continuous/semi-continuous processes (CSC) as the following mixed 0-1 linear programming problem, then this will facilitate our discussion and presentation of the smooth-and-dive accelerator.

$$\text{maximize } \mathbf{e}^T \mathbf{x} + \mathbf{f}^T \mathbf{y} - w \sum_{i=1}^{n_2} (y_i - y_i^2) \quad (1)$$

$$\text{subject to } \mathbf{Ax} \leq \mathbf{a} \quad (2)$$

$$\mathbf{By} \leq \mathbf{b} \quad (3)$$

$$\mathbf{Cx} + \mathbf{Dy} \leq \mathbf{c} \quad (4)$$

$$\mathbf{l} \leq \mathbf{x} \leq \mathbf{u} \quad (5)$$

$$\mathbf{y} \in \{0,1\} \quad (6)$$

where \mathbf{x} are the continuous or quantity variables representing the flows and inventories of size $n_1 \times 1$ subject to capacity bounds and \mathbf{y} are the binary or logic variables representing the modes, materials and moves for or between equipment of size $n_2 \times 1$. Objective function (1) combines both continuous and discrete variable economic, system performance and penalty (i.e., excess and deficit artificial variables) measures as well as the quadratic expression for encouraging integer-feasible solutions (i.e., drive fractional or pre-emptive \mathbf{y} variables to either 0 or 1) which will be described in more detail later. Constraints (2) to (4) contain such relations as the equipment material balances, operational logistics and any constraints involving both continuous and discrete variables such as semi-continuous flow modeling respectively. Bounds (5) are the upper and lower limits on each \mathbf{x} and bounds (6) are typically enforced by the implicit enumerative search when $w = 0$, or when the SDA is active, are relaxed to be $\mathbf{0} \leq \mathbf{y} \leq \mathbf{1}$ and are encouraged to be integer. The first to recognize that integer bounds (6) can be smoothed as the last term of constraint (1) was Raghavachari (1969) who described it as a concave quadratic program (QP) for which global optimization was

identified as the only approach to find the global optimum. Independently, Goldmann *et. al.* (1970) used this notion to derive production schedules using a QP formulation where their focus was on solving the switch-over minimization problem which did encourage integer-feasible solutions; this approach was also alluded to in Simon and Azma (1982). Of late, Karamarkar *et. al.* (1990) applied the idea of replacing the integrality restrictions on \mathbf{y} as another non-convex QP and solved difficult *set covering* problems using an interior-point programming approach with rounding. More recently, complementarity and smoothing functions applied to the $\max(0,y)$ and $\text{abs}(y)$ functions have been proposed to solve discontinuous optimization problems by Gopal and Biegler (1999) using the equations found in Chen and Mangasarian (1996). Other heuristic approaches, which are worth mentioning, are the interior-path heuristic found in Faaland and Hillier (1979) and the pivot-and-complement heuristic by Balas and Martin (1980). As well as two meta-heuristics, the first being tabu search which has been successfully used to enhance the pivot-and-complement primal heuristic by Aboudi and Jornsten (1994) and the second is GRASP, invented by Feo and Resende (1995) employing multiple restarts. These are not related to the SDA approach although aspects of each could be incorporated into the algorithm if required in the future. Specific to the SDA approach, we apply the smoothing techniques found below to encourage integer-feasible or near-integer-feasible solutions in order to speed up the processing of the discrete search and thus this leads us into the details of the SDA found in the next two sections.

Smoothing Functions applied to 0-1 Variables

In this section we detail four smoothing functions to enable the SDA to encourage integer-feasible solutions. The first smoothing function is the *quadratic* (QD) smoothing function proposed by Raghavachari (1969) and is modified as follows inspired by the other smoothing functions of next:

$$\sum_{i=1}^{n2} (y_i - y_i^2)^\beta = \sum_{i=1}^{n2} (y_i(1 - y_i))^\beta \quad (7)$$

The parameter β is perhaps mislabeled as a true smoothing parameter although it does influence the response surface between 0 and 1 for \mathbf{y} and it does allow us to generate different integer-feasible solutions for the same rim of the model and the same optimization settings. When $\beta = 1$ this corresponds exactly to the last term found in constraint (1). The second smoothing function, the *sigmoidal* (SG) smoothing function found in Chen and Mangasarian (1996), also known as the *neural network* smoothing function, is formulated to smooth the well-known sum of integer-infeasibility metric (see Balas and Martin, 1980 and Nemhauser and Wolsey, 1988):

$$\sum_{i=1}^{n2} \min(y_i, 1 - y_i) \quad (8)$$

When formulated into the SG smoothing function we get after little algebra:

$$\sum_{i=1}^{n^2} \left(y_i - \beta \ln \left(1 + \exp \left(\frac{-(2y_i)}{\beta} \right) \right) \right) \quad (9)$$

The third smoothing function is the *interior-point* (IP) smoothing function also known as the Chen-Harker-Kanzow-Smale function. It is written specifically to smooth equation (8) as:

$$\sum_{i=1}^{n^2} \left(y_i - \frac{1}{2} \left(2y_i - 1 + \sqrt{(2y_i - 1)^2 + \beta^2} \right) \right) \quad (10)$$

The fourth smoothing function is known as the *Fischer-Burmeister* (FB) smoothing function found in Fischer (1992) but is specific to the solution of complementarity problems and is not applied to equation (8). The FB function is used to allow optimization of

$$y_i \cdot (1 - y_i) = 0 \quad (11)$$

Applied to our problem we get:

$$\sum_{i=1}^{n^2} \left(1 - \sqrt{y_i^2 + (1 - y_i)^2 + \beta^2} \right) \quad (12)$$

Other such smoothing terms can be generated for example by maximizing

$$\sum_{i=1}^{n^2} \left(\text{abs} \left(y_i - \frac{1}{2} \right) \right) \quad (13)$$

which also tries to encourage integer-feasible solutions by pushing the \mathbf{y} variables as far away from 0.5 as possible hence driving towards either a 0 or 1 solution. Both the SG and IP can be easily applied to this term but for our purposes we concentrate on only equations (7), (9), (10) and (12). Moreover, it should be pointed out that there are other smoothing functions similar to SG and IP such as the *uniform* smoothing function and these too are not pursued further.

Smooth-and-Dive Accelerator Method

The SDA primal heuristic is mathematically uninteresting but has two major benefits when coupled with a branch-and-bound or branch-and-cut search. Both benefits are well-known and are related to the fact the SDA is intended to find integer-feasible solutions quickly where in some instances may converge to integer-feasible solutions without the need for the post search clean-up. When an integer-feasible solution is known, all of the active LP nodes in the search tree that have an objective function smaller than the incumbent integer-feasible solution can be pruned or fathomed hence reducing substantially the amount of memory required to support the search (value dominance). The other benefit is the ability to apply *reduced-cost fixing* after an integer-feasible solution has been found (Balas and Martin, 1980). This provides the search with a *combinatorial implosion* so to speak by fixing binary variables to either 0 or 1 if their reduced-cost from the root LP is greater than or equal to the difference between the root LP node objective function value and the incumbent integer-feasible solution's objective value. Unfortunately the success of

reduced-cost fixing is tenuous given that the reduced-costs available from the root LP relaxation may not be non-zero hence offering no binary variable fixes although with strong formulations and cutting-plane automatic reformulations the dual solution information may become more useful. Obviously a third and tacit benefit of the SDA is to simply find integer-feasible solutions to combinatorial hard optimization problems.

After the root LP relaxation is computed which may or may not include cutting-planes or valid inequalities, the first step of the SDA is to decide if the 0's, 1's, both or none of the LP binary variables that happen to be integer will be fixed for the next and subsequent iterations of the successive linear programming recursions of the SDA. Any fractional binary variables that are 0.5 can be randomly perturbed away from 0.5 if desired using a uniform distribution random number generator given that 0.5 offers no information in terms of the steepest descent direction. For the subsequent linear programs, the selected smoothing objective function type i.e., either (7), (9), (10) or (12), its non-negative parameter β and weight w are chosen and included in constraint (1); clearly the nonlinear smoothing functions described in the previous section must be linearized using a straightforward Taylor series expansion. The LP's are presolved and optimized using either the primal or dual simplex method or the barrier interior-point method until the last term of constraint (1) equals zero (a local optimum) or until no further improvement in the smoothing function can be obtained up to a maximum number of iterations. If the smoothing objective function does not converge to zero then either a branch-and-bound or branch-and-cut is initiated after all fractional binary variables are declared explicitly as *binary* in the global search of the optimizer and near-integer and integer binary variables are rounded to be exactly integer and fixed before the call. It is also possible to replace the implicit enumerative search with the simple and sometimes effective "dive-and-fix" primal heuristic found in Wolsey (1998) branching greedily on either the *best* or *worst* fractional binary variable one variable at a time. It may also be fruitful for certain problem instances to apply what is known as "randomized rounding" from Raghavan and Thompson (1987) after the SDA has converged to perform a probabilistic-type of rounding.

From the standpoint of generating other integer-feasible solutions, which can be considered as a poor man's local search, there are six potential avenues to explore. The first is to change the random seed when randomly perturbing the 0.5 binary variables at each iteration; note that randomization is a substitute for search memory albeit perhaps a poor one. The second is to use a different smoothing objective function type. The third is to change the smoothing parameter β within its range (this range should be experimented for the class of problem being solved). The fourth is to restart the SDA from a randomly perturbed root LP relaxation solution which may (or may not) push the search direction into more fruitful regions. The fifth is to after each integer-feasible solution has been found, to possibly add one or many incumbent elimination constraints (Aboudi and Jornsten, 1994) to the MILP formulation in order to cut-off the next run of the SDA from finding the same previous integer-feasible solutions which will serve to diversify the search space. And finally the sixth technique to persuade other integer-feasible solutions to be found is through the popular technique of selectively *dropping* constraints (as opposed to adding constraints as in the branch-and-cut

search), which have large associated slack variables. One set of constraints that are very suitable for constraint dropping in production scheduling problems are the semi-continuous flow variable upper and lower bounding constraints.

Moreover, for the production scheduling problems encountered in the hydrocarbon processing industries such as in oil-refineries and petrochemical complexes, the quality or property aspects i.e., the “intensive” variables of the production such as sulfur and octane levels, must then be taken into consideration after the quantity and logic aspects have been accounted for. This implies that even the best quantity-logic solutions found by the SDA may not yield the best quantity-quality production solution. A similar idea to this can be found in Glismann and Gruhn (2001) where separated short-term scheduling from recipe optimization.

Illustrative Example

The industrial-scale production scheduling problem exposed in this work is taken from the hydrocarbon processing industry. Specifically we apply the SDA to the oil-refinery business problem of mixing received crude-oils from either marine vessels or pipelines into storage tanks where there are typically more crude-oils than tanks (the classic *pigeon-hole principle* problem), transferring or moving the crude-oil mixes to the appropriate charge tank or tanks and then from these tanks moving the crude-oil mixture to the required crude-oil distillation pipestill being operated in a specific production mode. The mixed 0-1 linear programming model is formulated in discrete-time similar to the work of Lee *et. al.* (1996) and Shah (1996) including transition/switch-over minimization for the semi-continuous out-flows or moves between pipelines, tanks and pipestill; see Jia *et. al.*, (2002) for a continuous-time formulation. The basis of the formulation comprises the classic operation research (OR) problems known as the *fixed-charge network flow*, *facility-location* and *lot-sizing* problems with appropriate reformulations found in Sahinidis and Grossman (1991) and Wolsey (1998) where specific side-constraints are required to model the detailed intricacies of the crude-oil blendshop (i.e., mixing delays, standing-gauge tanks, one-flow-out of tank at a time, etc.). All experiments were modeled and programmed using *Xpress-Mosel* with all LP and MILP dual simplex optimizations using default settings performed by *Xpress-Optimizer* both available from Dash Optimization Inc. and were run on a laptop Pentium III 750 MHz computer. All SDA cases were executed using an arbitrary random seed of 1, a convergence tolerance of 0.01 for the QD smoothing objective function term (equation (10)), an integer tolerance of 0.001 and no binary variable fixing was performed in the SDA although fixing the 1's gave comparable results to those shown. The smoothing function weight w was ten times greater than the largest non-penalty or non-artificial variable objective function coefficient. Only the branch-and-cut search without SDA results were generated using temporal priorities or directives and used special-ordered-sets one (SOS1). These techniques were not used for the branch-and-cut search with SDA; it should also be mentioned that the branch-and-bound results offered similar overall trends to the branch-and-cut results and are not presented.

Table 1 displays the problem statistics without and with the technique known as presolve; all LP and MILP runs had presolve activated. Table 2 shows the results of the branch-and-cut search without SDA generating 5 integer-feasible solutions. This global

search has 1,320 binary variables to search over where the final objective function value of the best integer-feasible solution found of 253.9000 has a relatively small integrality gap compared to the root LP relaxation of 256.7474 which implies a reasonably well-formulated problem instance. The last column of Table 1 shows the QD value after the root LP has run of 19.2078. A very weak upper bound is $1320/4 = 330$ if all binary variables are at their worst position of 0.5.

Table 1. Problem Statistics including Presolve Results as the Last Row.

# Rows	# Columns	# Non-zeros	# 0-1 Vars.	# SOS1	Root LP OBJ Value	Root LP QD OBJ Value
10074	10962	32036	1320	682	256.7474	19.2078
4062	5463	12489	-	-	-	-

Table 2. Integer-Feasible Solution Results for Branch-and-Cut without SDA.

I-F Sol #	# Nodes	# Seconds	OBJ Value
1	240	9	-1486.7000
2	617	19	-906.6000
3	1071	29	-326.2000
4	2402	61	73.8000
5	3739	90	253.9000

Table 3 presents the main results of the paper where two different β values for each function are given in column two. As can be seen in column five (# 0-1 Vars.) there is a dramatic reduction in the number of binary variables used in the global search of over 96%! This as mentioned previously is the major advantage of adding to the global search the smooth-and-dive accelerator prefix. We can see in the last column that the first three smoothing methods QD, SG and IP, for an arbitrary choice of β , achieve nearly the same objective function value as the optimal branch-and-cut but in considerably less time (i.e., 90 seconds compared to 6 seconds). Unfortunately the FB function is not as beneficial for this problem as the others although it has performed better than the rest on other problem instances.

Table 3. Integer-Feasible Solution Results for SDA with Branch-and-Cut.

OBJ	Beta	# LP's	QD OBJ Value	# 0-1	# Nodes	# Seconds	# I-F Sol's	Best OBJ Value
OD	0.75	5	6.0921	33	10	5	1	253.8000
	1.75	10	5.1781	35	7	9	1	-276.3000
SG	0.05	7	7.3805	49	441	10	3	93.8000
	0.50	6	5.8428	32	11	5	1	253.8000
IP	0.05	5	7.2273	41	63	6	3	154.0000
	0.50	6	6.0655	36	39	6	3	253.8000
FB	0.00	30 (max)	11.8508	76	145	28	4	-2559.2000
	0.05	30 (max)	12.6354	75	168	27	3	-3289.5000

In addition, the SDA prefix to the branch-and-cut search has found 12 integer-feasible solutions of better quality in 41 seconds when we add up the total time taken to solve the problem using QD, SG and IP whereas the branch-and-cut alone found only 3 integer-feasible solutions in the same amount of time of lesser quality.

Conclusion

The smooth-and-dive accelerator primal heuristic described in this short note has been successfully applied to a large-scale blendshop scheduling problem found in the hydrocarbon processing industry. The reduction in the number of 0-1 variables from the root linear programming relaxation to the invocation of the branch-and-cut is the main benefit of the SDA and has demonstrated clear value over the standalone branch-and-cut implementation. Since the SDA is an optimization procedure in itself, it is complementary to the well-established and elegant theory of integer programming in that all aspects of valid inequalities, extended formulations and problem tightening can enhance the success of solving any hard combinatorial problem formulated and solved in the MILP framework. Likewise, decomposition strategies such as Lagrangean relaxation, Bender's decomposition and other decomposition strategies such as the pragmatic temporal decomposition of Bassett *et. al.* (1996) can all be easily augmented by the SDA heuristic. The SDA can also be easily set-up in a parallel computing environment initiated with different random seeds and/or using different smoothing objective function types and parameters. This has the advantage of generating many integer-feasible solutions on several computers simultaneously from which the user can choose which production scheduling solution to implement which best suits the manufacturing environment given the current situation or state of the plant. Finally, it should also be mentioned that several of the other OR problems found in Rardin (1998) such as the *0-1 knapsack*, *traveling salesman* and *set partitioning* problems have been tried with similar results to the illustrative example although these problem instances are not of any interesting practical scale.

References

- Aboudi, R. and Jornsten, K., "Tabu search for general zero-one integer programs using the pivot-and-complement heuristic", *ORSA Journal on Computing*, **6**, 1, 82-93, (1994).
- Balas, E. and Martin, C.H., "Pivot and complement – a heuristic for 0-1 programming", *Management Science*, **26**, 1, 86-96, 1980.
- Bassett, M.H., Pekny, J.F., and Reklaitis, G.V., "Decomposition techniques for the solution of large-scale scheduling problems", *AIChE Journal*, **42**, 12, 3373-3387, (1996).
- Chen, C. and Mangasarian, O.L., "A class of smoothing functions for nonlinear and mixed complementarity problems", *Comput. Optim. Appl.*, **5**, 2, 97-138, (1996).
- Dash Optimization Inc., *Xpress-Optimizer Reference Manual*, Release 2003, (2002).
- Faaland, B.H. and Hillier, F.S., "Interior path methods for heuristic integer programming procedures", *Operations Research*, **27**, 6, 1069-1087, (1979).
- Feo, T.A., and Resende, M.G.C., "Greedy randomized adaptive search procedures", *Journal of Global Optimization*, **6**, 109-133, (1995).
- Fischer, A., "A special Newton-type optimization method", *Optimization*, **24**, 269-284, (1992).
- Glismann, K. and Gruhn, G., "Short-term scheduling and recipe optimization of blending processes", *Computers chem. Engng.*, **25**, 627-634, (2001).
- Goldmann, S.F., Griffiths, J.R., Perez, M. and Sinai, J., "Operations scheduling – a quadratic programming approach", *Esso Mathematics & System Inc.*, Report No. **EMS. 11M70**, (1970).
- Gopal, V. and Biegler, L.T., "Smoothing methods for the treatment of complementarity conditions and nested discontinuities in chemical process engineering", *AIChE Journal*, **45**, 7, 1535-1547, (1999).

- Jia, Z., Ierapetritou, M. and Kelly, J.D., "Refinery short-term scheduling using continuous-time formulation – crude-oil operations", submitted to *Industrial & Engineering Chemistry Research*, February, (2002).
- Karamarkar, N., Resende, M.G.C., and Ramakrishnan, K.G., "An interior point algorithm to solve computationally difficult set covering problems", Mathematical Sciences Research Center, AT&T Bell Laboratories, Murray Hill, N.J., August, (1990).
- Lee, H., Pinto, J.M., Grossmann, I.E., and Park, S., "Mixed-integer linear programming model for refinery short-term scheduling of the crude oil unloading with inventory management", *Industrial Engineering Chemistry Research*, **35**, 5, 1630-1641, (1996).
- Nemhauser, G.L and Wolsey, L.A., *Integer and Combinatorial Optimization*, John Wiley & Sons, New York, (1988).
- Raghavachari, M., "On connections between zero-one integer programming and concave programming under linear constraints", *Operation Research*, **17**, 680-684, (1969).
- Raghavan, P. and Thompson, C.D., "Randomized rounding: a technique for provably good algorithms and algorithmic proofs", *Combinatorica*, **7**, 4, 365-374, (1987).
- Rardin, R.L., *Optimization in Operations Research*, Prentice Hall, Engle-Wood Cliffs, N.J., (1998).
- Sahinidis, N. V. and Grossman, I.E., "Reformulation of multiperiod MILP models for planning and scheduling of chemical processes", *Computers & Chemical Engineering*, **15**, 4, 255-272, (1991).
- Shah, N., "Mathematical programming techniques for crude oil scheduling", *Computers & Chemical Engineering*, **20**, Suppl. B, S1227-S1232, (1996).
- Simon, J.D. and Azma, H.M., "Exxon experience with large scale linear and nonlinear programming applications", *Computers & Chemical Engineering*, **7**, 5, 605-614, (1983).
- Wolsey, L.A., *Integer Programming*, John Wiley & Sons, New York, (1998).