.

AURORA DEL CAMP – SPECIFICATION

JULIAN PFEIFLE

This document outlines some of the issues that arise in the implementation of Gilad's proposal [2].

Given the availability of powerful free and open source solvers for integer programs such as CBC [3], it seems natural to pursue an integer programming formulation. Of course, free solvers are not as good as the best commercial ones, but the most recent benchmarks [4] indicate that CBC is reasonably competitive; more precisely, it's the most competitive among all solvers that have an open source license (in the case of CBC, the "Eclipse Public License") that permits Gilad to use it commercially without paying any license fees.

1. Approaches to modeling

Before summarizing some approaches to modeling the problem, we briefly list the activities involved.

1.1. Tasks to be considered.

The tasks relevant to the problem come in two types, those that only affect the field, and are thus common to all crops, and those that are specific to each crop.

Tasks common to all crops: ti for tilling, rv for rotovating, gm for green manure planting, ft for fertilizing, bb for bed building, si for setting up irrigation, sr for setting rows, we for weeding.

Tasks specific to a crop: by for buying seeds, ss for soaking seeds, cs for cutting or separating cloned seeds, gc for false germination and cleaning, pl for planting[1], fu for fumigating, th for thinning, tr for trimming, co for covering, ha for harvesting.

We record the precedence constraints between these tasks in a directed acyclic graph E. Thus, $t_1 \longrightarrow t_2$ (also written $t_1 < t_2$) is a directed edge in E if t_1 must be completed before t_2 can start.

1.2. Approaches.

We consider two types of models:

Discrete time, all-in-one: The first model discretizes time into 52 weeks, and reserves a 0/1 variable for each triple in $T_c \times W \times A$, respectively quadruple in $T_s \times C \times W \times A$. Here $T = T_c \cup T_s$ denotes the set of common and specific tasks, C the set of crops, W the set of weeks in the planning horizon of the model (so, for example, $W = \{0, 1, \ldots, 51\}$ corresponds to modelling one year), and A the set of *unit lots* that the available farmland is divided into. Each variable, say $v_{t,w,a}$, takes the value 1 (respectively, 0), if the task t is (respectively, is not) executed in week w in the unit lot a. Thus, this model simultaneously solves the *scheduling* part of the problem, in which the appropriate sequencing of events is determined, and the *allocation* part, in which tasks and crops are assigned their proper place in the field.

Event-based, separate: The second model separates these two parts of the problem. First, the optimal sequencing of events is determined in a way that respects the available field space and work force using an *event-based* formulation (see below); in a second step, each task is allocated its proper space in the field. Separating the two phases makes it easier to formulate each one, and presumably makes them (and thus, the whole problem) easier to solve.

Date: Version of November 23, 2011.

[1] We consider transplanting and planting to be the same process.

Since the event-based strategy appears to be more powerful, we describe it first; the discrete-time formulation might eventually disappear completely from this document.

2. Problem formulation using events

Here we follow [1]. Put briefly, the resources consumed are time, money and space in the fields, and money is produced again by selling the crops.

3. Problem formulation using discrete time

This kind of formulation needs significantly more variables, but has a better LP-relaxation. Only by testing both will we be able to determine which is the superior method for our problem. To keep the number of variables somewhat managable, we work with *circular time* by discretizing one year into 52 weeks, and identifying the start and end of that year.

3.1. **Variables.** We work with a set C of crops, the set $W = \{0, \dots, 51\}$ of weeks in the year, and a set A of "unit lots", i.e., indivisible units of farmland dedicated to a single crop or task. The set A has a distinguished member $a_0 \in A$ that stands for off-field work. In this fictitious lot, any number of tasks may be performed simultaneously, while only one thing at a time may be done in all lots $A \setminus a_0$.

In the discrete time formulation, each of the "common" families contains variables indexed by the week $w \in W$ of the year, and the piece of land $a \in A$. For instance, the tilling variables are $ti = \{ti_{w,a} : w \in W, a \in A\}$. The "crop-specific" families, on the other hand, contain more variables, because they also depend on the type of crop $c \in C$. For example, the set of seed-soaking variables is $ss = \{ss_{c,w,a} : c \in C, w \in W, a \in A\}$. Each individual variable in each of these families can take on the value 0 or 1, depending on whether or not the task at hand is undertaken for crop c in the unit lot a during week w.

With an estimated 40 types of crops and 40 unit lots, we get an upper bound of

$$8 \times 40 \times 52 \text{ (common)} \;+\; 10 \times 40 \times 40 \times 52 \text{ (specific)} \;=\; 848\,640 \text{ variables.}$$

One the one hand, this is well within the reach of commercial solvers such as Gurobi (we'll have to see how cbc performs, though); on the other, there will actually be substantially less variables than this because not all crops need all variables. For example, a crop that is bought as a seedling from a nursery does not need variables from the families by, ss, cs, gc.

3.2. **Constraints.** To formulate our constraints, we need some additional data on our crops:

Plantation interval pi_c: How many weeks must pass, at least, between plantings of crop c
Yield $y_{c,i}$: The amount of fruit that crop c yields i weeks after planting, for $0 \le i \le pi_c$.

Now we can formulate the constraints:

(1) *In each week, there is at most one crop planted at each unit lot:*

$$\sum_{c \in C} pl_{c,w,a} \;\le\; 1 \qquad \text{for all } w \in W \text{ and } a \in A$$

(2) *Respect plantation intervals:*

$$\sum_{i=w}^{w+pi_c} pl_{c,i,a} \;\le\; 1 \qquad \text{for all } w \in W, c \in C, a \in A, \tag{3.1}$$

because during any interval of pi_c consecutive weeks, there may occur at most one planting. Here and throughout, index additions such as $i + pi_c$ are understood to be taken modulo 52.

(3) *Prescribe the yield:* If we want the total yield of crop c in a given week $w \in W$ to be $Y_{c,w}$, we must impose

$$\sum_{a \in A} \sum_{i=0}^{pi_c} y_{c,i}\ pl_{c,w-i,a} \ = \ Y_{c,w},$$

because a crop planted i weeks before week w has a yield of $y_{c,i}$ in week w. This works because by the foregoing constraint (3.1), at most one of the variables $\{pl_{c,w-i,a} : 0 \leq i \leq pi_c\}$ can have the value 1, while the rest must be 0.

(4) *Precedence constraints:* Here we must deal with the fact that we model our year to be cyclic, i.e., the same 52 weeks repeat again and again, as witnessed by our index addition modulo 52 in condition (3.1). Nevertheless, some tasks must be completed before others can start, which implies a linear and not circular ordering of time. We model this by interpreting precedence constraints to be valid only within a half-year horizon, so that *"x must happen before y"* is interpreted as meaning *"x may not happen until half a year after y"*. For example, the constraint that "tilling" must happen before "weeding" for each crop is modeled as

$$\begin{aligned} ti_{w+i,a} \ &\leq \ we_{w,a} && \text{for all } w \in W,\ a \in A,\ \text{and } i \in \{0, \ldots, 26\}, \\ ti_{w+i,a} + we_{w,a} \ &\leq \ 1 && \text{for all } w \in W,\ a \in A,\ \text{and } i \in \{0, \ldots, 26\}. \end{aligned}$$

To check this, note that these inequalities are not satisfied, for example, if $ti_{4,c} = 1$ and $we_{0,c} = 1$ (which is as it should be, because these variables say we till in week 4, after having weeded in week 0), but they are satisfied if $ti_{0,c} = 1$ and $we_{4,c} = 1$.

3.3. **Objective function.** Each crop $c \in C$ has a yield of $y_{c,w}$, depending on the week $w \in W$ it is planted. The objective function we want to maximize is thus

$$f \ = \ \sum_{c \in C, y \in Y} y_{c,w} \sum_{a \in A} x_{c,w,a}$$

4. Server-side technology

Gilad's intention is to make the program available on a server. That's fine, except that we need to be able to install c++ and cbc on such a server.

References

[1] C. Artigues, O. Koné, P. Lopez, and M. Mongeau, *Comparison of mixed integer linear programming models for the resource-constrained project scheduling problem with consumption and production of resources*. `http://www.math.univ-toulouse.fr/~mongeau/MILP-RCPSP-CPR-submitted.pdf`, February 2011.

[2] G. Buzi, *Aurora Del Camp Crop Planner — a small farm plans big*, November 2011.

[3] *Cbc (coin-or branch and cut), an open-source mixed integer programming solver written in C++*. `https://projects.coin-or.org/Cbc`.

[4] H. D. Mittelmann, *Performance of optimization software — an update*. `http://plato.asu.edu/talks/mittelmann_bench.pdf`, November 2011.