A Branch-and-Price-and-Cut approach for Sustainable Crop Rotation Planning

Laurent Alfandari, Agnès Plateau, Xavier Schepler

A Branch-and-Price-and-Cut Approach for Sustainable Crop Rotation Planning

Research Center
ESSEC Working Paper 1408

2014

Laurent Alfandari
Agnès Plateau
Xavier Schepler

ESSEC
BUSINESS SCHOOL

A Branch-and-Price-and-Cut approach for Sustainable Crop Rotation Planning

Laurent Alfandari[a], Agnès Plateau[b], Xavier Schepler[c]

[a]*ESSEC, IDS*
avenue Bernard Hirsch, BP 50105
95021, Cergy, France
[b]*CNAM, CEDRIC*
292, rue Saint Martin
75141, Paris Cedex 03, France
[c]*Normandie Université, LITIS, LMAH*
25, rue Philippe Lebon, BP 540
76058, Le Havre, France

Abstract

In this paper, we study a multi-periodic production planning problem in agriculture. This problem belongs to the class of crop rotation planning problems, which have received increased attention in the literature in recent years. Crop cultivation and fallow periods must be scheduled on land plots over a given time horizon so as to minimize the total surface area of land used, while satisfying crop demands every period. This problem is proven strongly NP-hard. We propose a 0-1 linear programming compact formulation based on crop-sequence graphs. An extended formulation is then provided with a polynomial-time pricing problem, and a Branch-and-Price-and-Cut (BPC) algorithm is presented with adapted branching rules and cutting planes. The numerical experiments on instances varying the number of crops, periods and plots show the effectiveness of the BPC for the extended formulation compared to solving the compact formulation, even though these two formulations have the same linear relaxation bound.

Keywords: OR in agriculture, crop rotations, production planning, column generation, branch-and-price-and-cut

March 26, 2014

1. Introduction

Although definitions of sustainable agriculture may vary, agricultural systems are generally considered as sustainable if they sustain themselves along a long period of time, that is, if they are economically viable, environmentally safe, and socially fair. In particular, sustainable agricultural practices are usually requested to incorporate alternatives to toxic fertilizers and pesticides, avoid excessive tillage and preserve soils. Many research papers about sustainable agriculture focus on the pollution and social side-effects of intensive agriculture, such as water spoiled by pesticides, crop diseases, and concentration of production in fewer and bigger farms that can afford large investments in costly automative production systems and technologies (e.g., [1; 2]). Consistent with this diagnosis, recommendations on the need for new sustainable agricultural systems can be found in [3]. Crop rotations, combined with fallow periods where the land rests in order to recover its soil attributes after production, enable crop diversification on both space and time dimensions. Typical crop rotation problems usually focus on building rotations that maximize a profit or yield function, where the total surface area of land is either unbounded or fixed [4; 5; 6; 7]. This paper deals with an aspect of sustainability which is rarely considered in optimization of agricultural production systems: the minimization of the surface area needed to cover crop demands that vary over time. A compact formulation for a mixed-integer variant of the problem was originally introduced in [8], following a communication in the EURO XXI Conference in 2006. Since then, a number of papers have addressed crop rotation planning in a sustainable development context. For example, a column generation approach was applied in [7], where the objective is to maximize space occupation and the master problem includes adjacency constraints between plots. Column generation was also used in [9] for a crop rotation problem with land divided into plots, but with continuous variables representing the surface area assigned to a given rotation, hence requiring no branching. Another example can be found in [10] where harvested crops can be stocked for a limited period of time, and demands are subject to uncertainty. [11] presents multi-objective crop rotation models that take risk into account, converted into linear programs and solved with standard Linear Programming (LP) methods. A survey on crop rotation decisions exists [12] but does not include most recent papers in optimization. Dantzig-Wolfe decomposition [13] was applied to our problem in [14], but with no inclusion in a branch-and-price approach to obtain opti-

mal integer solutions. To our knowledge no branch-and-price algorithm has ever been designed so far for any crop rotation planning problem.

The original model presented in [8] was motivated by a Madagascan case study where the minimization of cultivated space contributed to the sustainable development of the primary forest in the long term. Indeed, farmers in Madagascar are used to clearing more and more primary forest areas - although this is prevented by law - in order to extend their cultivation surface area to better cover their needs. A plot could be cultivated with several crops in the same period in this study. We direct the reader to [8] for more details on the agricultural Madagascan context. In this paper, we present a fully-combinatorial problem where a single crop can be cultivated on each plot at each period.

The paper is organized as follows. Section 2 introduces notation and crop-sequence graphs. Section 3 describes a compact formulation of the problem. Section 4 proves NP-hardness. Section 5 provides a Covering Integer Programming extended formulation derived from a Dantzig-Wolfe decomposition approach. Section 6 presents the Branch-and-Price-and-Cut with branching rules and cutting planes. Section 7 presents computational experiments for various time horizons, number of crops and plot sizes. Section 8 concludes the paper.

2. Notations and crop-sequence graphs

We consider the following notations for the Minimum-Space Crop Rotation Planning problem ($MSCRP$):

- $t = 1, \ldots, T$: the periods of the planning horizon.
- $p = 1, \ldots, P$: the set of land plots that can possibly be used
- C : the set of crops, with $f \in C$ the fallow index considered as a specific crop for modeling reasons
- $C_t \subseteq C$: the set of crops that can be cultivated at period t
- d_{ct} : the demand (in tons) of crop $c \in C_t \setminus \{f\}$ at period t
- L : the number of fallow periods after which the yield no longer increases

State v	$Succ(v)$	$Pred(v)$
$(rice, l, 1)$	$(bean, l, 2)$, $(f, 1, 0)$	$(f, l, 0)$
$(rice, l, l')$, $1 < l' < L'$	$(bean, l, l' + 1)$, $(f, 1, 0)$	$(bean, l, l' - 1)$
$(rice, l, L')$	$(f, 1, 0)$	$(bean, l, L' - 1)$
$(f, l, 0)$	$(f, \min(l + 1, L), 0), (rice, l, 1), (bean, l, 1)$	$(rice, l, l'), (bean, l, l')$

Figure 1: Table of successors and predecessors of a state with two alternating crops

- L' : the maximum number of consecutive periods a plot can be cultivated before returning fallow
- s^p : the surface area of plot p (in ha)

The *state* of a plot p is a triplet $v = (c, l, l')$ where c is the crop (or fallow) at the current period, $l \leq L$ is the *fallow length*, i.e., the number of consecutive fallow periods before cultivation (if this number is greater than L then it is replaced by L), and $l' \leq L'$ is the *cultivation length*, i.e. the number of consecutive cultivation periods up to the current period. The only possible states are $(f, l, 0)$ for $l = 1, \ldots, L$, and (c, l, l') for $c \in C \setminus \{f\}$, $l = 1, \ldots, L$, $l' = 1, \ldots, L'$. When the plot is cultivated in period t and remains cultivated in the next period $t + 1$ the cultivation length l' of the plot is increased by one. When the maximum length of cultivation L' is reached, the plot has to return fallow, with fallow length $l = 1$ and cultivation length $l' = 0$. When a plot has been left fallow for l periods, it can either remain fallow the next period with length $\min(l + 1, L)$ or go back to cultivation with some crop c, fallow length l and cultivation length $l' = 1$. We denote by $Succ(v)$ the set of possible successors of state v at the next period, and by $Pred(v)$ the set of predecessors of state v at the previous period. Figure 1 provides the list of possible successors and predecessors of each state if $C = \{rice, bean, f\}$, rice precedes bean and bean precedes rice.

We denote by V_{pt} the set of possible states of plot p at period t. At the beginning of the planning horizon, V_{p0} is reduced to a single state $start_p$. We note for $t = 1, \ldots, T$

$$A_{pt} = \{(v, v') \in V_{p,t-1} \times V_{pt} : v' \in succ(v)\}$$

the set of possible *transitions* from a state $v \in V_{p,t-1}$ to a state $v' \in V_{pt}$, as illustrated by Figure 1. We also note, for each state $v \in V_{pt}$, $A^+_{pt}(v) =$

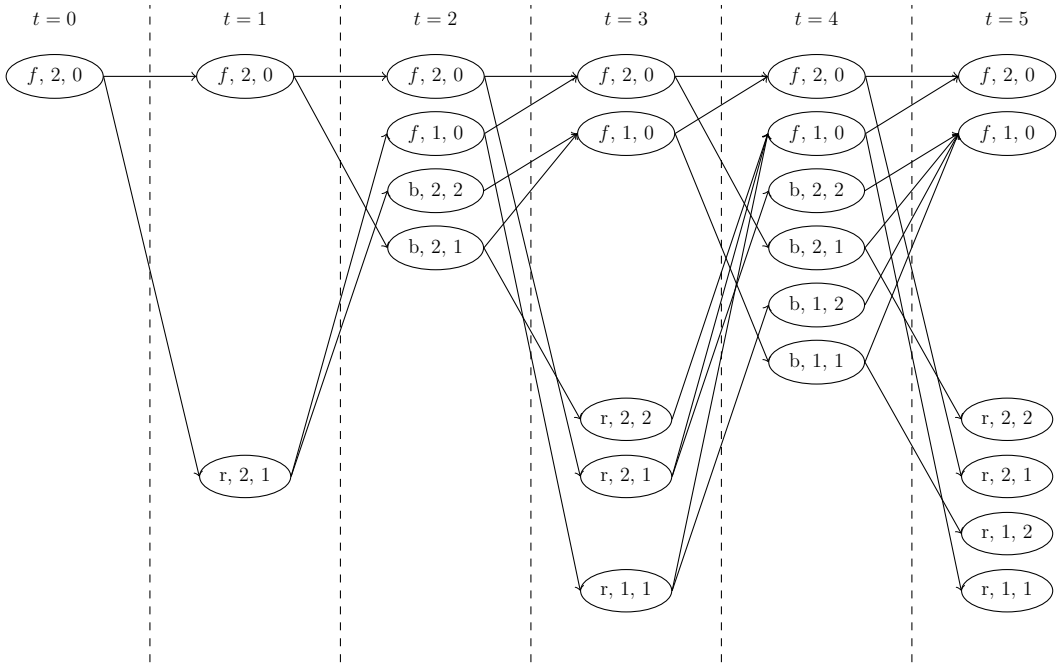

Figure 2: Crop-sequence graph

$\{(v, v') : (v, v') \in A_{p,t+1}\}$ and $A^-_{pt}(v) = \{(v', v) : (v', v) \in A_{pt}\}$ the set of transitions that start at state v and end at state v at period t, respectively.

Now, consider the acyclic directed graph $G^p = (V^p, A^p)$ with $V^p = \cup_{0 \leq t \leq T} V_{pt} \cup \{end_p\}$, where node end_p represents the end of a rotation, and $A^p = \cup_{1 \leq t \leq T} A_{pt} \cup \{(v, end_p) : v \in V_{pT}\}$. We call this graph the *crop-sequence graph*. By construction, any path from $start_p$ to end_p in graph G^p identifies a feasible crop rotation on plot p. For each crop $c \in C_t \setminus \{f\}$, we call $A^c_{pt} \subset A_{pt}$ the set of arcs such that crop c is cultivated at the final endpoint of transition a. Each arc $a \in A^c_{pt}$ is valued by $s_p w_{pac}$, where w_{pac} is the number of tons of crop c obtained by transition a on one hectare of plot p. All other arcs, i.e., those which have a fallow state $(f, l, 0)$ as final endpoint and those that terminate at end_p, have a zero value. Figure 6 describes such a crop-sequence graph for two possible crops rice (r) and bean (b) and two periods. The following section describes a compact formulation of the problem.

Figure 2 describes such a crop-sequence graph for two possible crops rice (r) and bean (b) and five periods. The following section describes a compact formulation of the problem.

3. Compact formulation

The Minimum-Space Crop Rotation Planning ($MSCRP$) problem is that of constructing crop rotations minimizing the total space area required for covering seasonal crop demands. We introduce the following Compact Formulation (CF) for $MSCRP$.

$$\min \quad \sum_{p=1}^{P} \sum_{a \in A_{p1}} s_p x_{pa1} \tag{1}$$

$$\text{s.t} \quad \sum_{p=1}^{P} \sum_{a \in A^c_{pt}} s_p w_{pac} x_{pat} \geq d_{ct} \qquad \forall c \in C_t \setminus \{f\}, t = 1, \ldots, T \tag{2}$$

$$(CF) \quad \sum_{a \in A^-_{pt}(v)} x_{pat} = \sum_{a \in A^+_{pt}(v)} x_{pa,t+1} \qquad \begin{array}{l} \forall p = 1, \ldots, P,\ t = 1, \ldots, T-1, \\ v \in V_{pt} \end{array} \tag{3}$$

$$x_{pat} \in \{0, 1\} \tag{4}$$

Binary decision variable x_{pat} is equal to one if and only if plot p uses transition a in period t. The objective function (1) minimizes the total surface area of plots p that are used for production, i.e. such that $\sum_{a \in A_{p1}} x_{pa1} = 1$. Global constraints (2) ensure that the total production of a crop is at least the demand for every period. Flow conservation constraints (3) are local constraints associated with a plot p and define a path structure for a rotation on that plot. Note that if $\sum_{a \in A_{p1}} x_{pa1} = 0$, for this plot p all variables x_{pat} are equal to zero which means that no crop rotation is used on that plot. The linear relaxation of CF will be noted $\overline{CF}$. We study the complexity of the $MSCRP$ problem in the following section.

4. Problem complexity

We prove the NP-hardness of this problem with a polynomial reduction from the (unweighted) Set Covering Problem (SCP). In the unweighted SCP, we are given a set of elements $I = \{1, \ldots, n\}$ and a collection of subsets of elements $\mathcal{S} = \{S_1, \ldots, S_m\}$ such that $S_j \subset I$. The objective is to find

a collection $\mathcal{S}' \subset \mathcal{S}$ such that $\bigcup_{S_j \in \mathcal{S}'} S_j = I$ and the size $|\mathcal{S}'|$ of the cover is minimum.

Theorem 1. *$MSCRP$ is strongly NP-hard and reduces to the Set Covering Problem (SCP).*

Proof: We transform a general SCP instance into a specific $MSCRP$ instance in the following way. Take $P = m$, $T = m + n$, $C = \{f, 0\} \cup I = \{f, 0, 1, \ldots, n\}$, $s^p = 1$ and $start_p = (f, L, 0)$ for all $p = 1, \ldots, P$. Moreover set $C_t = \{f, 0\}$ for $t = 1, \ldots, m$ and $C_{m+c} = \{c, 0\}$ for all $c = 1, \ldots, n$; crop c can only be produced in period $m + c$ and no other crop but crops c and 0 can be produced in that period. We only set demands for crops of $C \setminus \{f, 0\} = \{1, \ldots, n\}$, with $d_{c,m+c} = 1$ for each crop $c = 1, \ldots, n$. These are the only demands to be covered, in particular there are no demands to cover in periods $1, \ldots, m$. Also, set $L = L' = m + n$. Finally, we set all crop yields equal to zero, except the following yields for $c = 1, \ldots, n$, $j = 1, \ldots, m$:

$$\bar{w}^{c-1,c}_{L,j+c} = 1 \quad \text{iff} \quad c \in S_j, c - 1 \in S_j \tag{5}$$

$$\bar{w}^{0,c}_{L,j+c} = 1 \quad \text{iff} \quad c \in S_j, c - 1 \notin S_j \tag{6}$$

where, for sake of simplicity, we change yield notations as $\bar{w}^{c',c}_{L,j+c} = w_{pac}$ with a the transition from state $(c', L, j + c - 1)$ to state $(c, L, j + c)$. We now show the NP-completeness of the decision version of $MSCRP$. In order to do this, we show that if there exists a set cover of cost of no more than K in the SCP instance, then there exists a crop rotation plan of cost of no more than K in the $MSCRP$ instance, and vice-versa. Let us show the first implication.

(i) Assume that there exists a collection of no more than K subsets covering I in the SCP instance. For each subset S_j of this cover, we transform it into a rotation j on a plot for $MSCRP$ in the following way:

- for $t = 1, \ldots, m - j$, the plot is left fallow (f),
- for $t = m - j + 1, \ldots, m$, the plot is cultivated with crop 0,
- for $t = m + 1, \ldots, m + n$, in period $m + c$ $(c = 1, \ldots, n)$, if $c \in S_j$ then crop c is cultivated, otherwise crop 0 is cultivated.

If $c \in S_j$, then in rotation j and in period $m+c$ crop c is cultivated, and the state of the plot is $l = L$ and $l' = m + c - (m - j) = j + c$ in this period. Therefore by (5,6) the yield of crop c is 1 so the demand $d_{c,m+c} = 1$ in period $m + c$ is covered. As for all $c = 1, \ldots, n$ there exists at least one subset S_j in the SCP cover such that $c \in S_j$, all demands $d_{c,m+c} = 1$ are covered. Since the rotations so built are in one-to-one correspondence with the subsets of the SCP cover, we have no more than K plots used for covering the crop demands.

(ii) Conversely, assume that there exists a set of K rotations in the $MSCRP$ instance covering the crop demands for $t = m + 1, \ldots, m + n$. Denote by J the set of indices $j \in \{1, \ldots, m\}$ such that at least one demand $d_{c,m+c}$ is covered, in the $MSCRP$ solution, by a yield satisfying:

$$\bar{w}^{c-1,c}_{L,j+c} = 1 \text{ or } \bar{w}^{0,c}_{L,j+c} = 1 \tag{7}$$

As for each $j \in J$ a distinct plot should have started to be cultivated from period $t = m - j + 1$ and we have no more than K plots, then $|J| \leq K$. Moreover as the K rotations form a feasible solution for $MSCRP$, then for all $c = 1, \ldots, n$ there exists $j \in J$ satisfying (7), i.e., such that $c \in S_j$. So, to transform a $MSCRP$ solution of size K into a SCP cover of size not more than K, simply select subsets S_j for $j \in J$. This concludes the NP-completeness proof. As the above transformation of a general SCP instance into a particular $MSCRP$ instance is a pseudo-polynomial transformation (see [15]), $MSCRP$ is NP-hard in the strong sense, i.e., there exists no exact algorithm that solves the problem in a time which is polynomial in both the size of the instance and the largest integer of the input data, unless $P = NP$.

4.1. Extended formulation and column generation

We denote by $\mathcal{R}_p$ the set of feasible crop rotations for plot $p = 1, \ldots, P$ over the time horizon, and $\mathcal{R} = \cup_p \mathcal{R}_p$. The compact formulation CF (1-4) of section 2 can be reformulated as the following Master Problem (MP) after a Dantzig-Wolfe decomposition [13].

$$\min \quad \sum_{p=1}^{P} s_p \sum_{r \in \mathcal{R}_p} \lambda_r \tag{8}$$

$$\text{s.t} \quad \sum_{p=1}^{P} \sum_{r \in \mathcal{R}_p} w_{pct}^r \lambda_r \geq d_{ct} \quad \forall t = 1, \ldots, T,\ c \in C_t \setminus \{f\} \tag{9}$$

$$(MP) \qquad \sum_{r \in \mathcal{R}_p} \lambda_r \leq 1 \qquad \forall p = 1, \ldots, P \tag{10}$$

$$\lambda_r \in \{0, 1\} \qquad \forall r \in \mathcal{R} \tag{11}$$

where binary variable $\lambda_r = 1$ if and only if rotation $r \in \mathcal{R}$ is selected in the solution and w_{pct}^r is the number of tons of crop c produced by rotation r on plot p at period t.

The well-known iterative principle of a column generation approach [16] can be summarized as follows on the $MSCRP$. Let $\overline{MP}$ denote the linear relaxation of MP. The aim is to solve $\overline{MP}$ to optimality and get a lower bound which has the same value as the solution of the dual problem given by the Lagrangian relaxation of constraints (2) in the compact formulation (1-4) [17]. One starts to solve a Restricted Master Problem (RMP), i.e. problem $\overline{MP}$ restricted to a small subset of rotations (columns), by the simplex algorithm. This LP-solving of the RMP provides dual variables u_{ct} associated with the covering constraints (9) and dual variables u'_p associated with constraints (10). Then one checks whether there exists a rotation $r \in \mathcal{R}$ with negative reduced cost that could be added to the RMP in order to improve the LP bound. If no such negative reduced cost column exists, then the current solution of the last RMP is optimal for $\overline{MP}$, otherwise one adds a subset of negative reduced-cost columns to the RMP and reiterates the process until no negative reduced cost column is found.

At an iteration of the above Column Generation process, the subproblem of finding a column with minimum reduced cost is called the *pricing* problem. Given the current dual variables u_{ct} and u'_p output by the LP-solving of the last RMP, the reduced cost of a column (or rotation) $r \in \mathcal{R}_p$ is:

$$\overline{c}_r = s_p - \sum_{t=1}^{T} \sum_{c \in C_t \setminus \{f\}} w_{pct}^r u_{ct} - u'_p$$

The pricing problem of minimizing $\overline{c}_r$ over all rotations $r \in \mathcal{R}$ can be efficiently solved, despite the exponential size of $\mathcal{R}$, by running a shortest-path

dynamic programming procedure in each graph G^p. Given that a node in G^p is a state (c, l, l') for each season $t = 1, \ldots, T$, the number of nodes in G^p is in $O(|C|LL'T)$. As each graph G^p is acyclic, one can use Bellman algorithm to find a shortest path in that graph. To minimize $\bar{c}_r$ over all possible paths $r \in \mathcal{R}_p$, simply associate cost $-s_p w_{pac} u_{ct}$ to each arc (transition) a that produces crop c at period t (with $w_{paf} = 0$ if the arc ends at a fallow), and associate cost $s_p - u'_p$ to each arc whose initial endpoint is $start_p$, and a zero cost to each arc whose terminal endpoint is end_p.

At the end of the column generation algorithm, the value of the last RMP is the value of the linear relaxation of MP. One could run a MIP solver on the subset of columns of this last RMP, but this does not ensure optimality of the integer solution. This heuristic, noted HEUR, will be compared to the exact methods in the numerical experiments section. Branch-and-price is a branch-and-bound method which finds an optimal integer solution, if no time limit is fixed, using the above column generation algorithm to compute the LP bound at each node. We refer to [18; 19] for a detailed description of branch-and-price. Branch-and-Price-and-Cut (BPC) is an enriched branch-and-price where cutting planes are iteratively added to accelerate the solving of the master problem. In the next subsections we describe the branching rules used for our problem as well as the cut generator.

4.2. Branching scheme

Branching occurs at the end of the processing of a node from the search tree when the node cannot be pruned, i.e., the obtained lower bound is not over any known upper bound, and the solution of that node, noted $\tilde{\lambda}$, is fractional. Then there exists a plot p, a period t and an arc a such that:

$$0 < \sum_{\substack{r \in \hat{\mathcal{R}}_p: \\ x^r_{pat}=1}} \tilde{\lambda}_r < 1$$

where $\hat{\mathcal{R}}_p$ is the set of columns on plot p in the last RMP, and $x^r_{pat} = 1$ if transition a is used at period t in rotation $r \in \hat{\mathcal{R}}_p$, 0 otherwise. The arc a on which branching will be performed is selected in a set A_{pt} as follows. First, the smallest t is selected such that there exists a plot p and an arc $a \in A_{pt}$ providing a fractional value of $\sum_{r \in \hat{\mathcal{R}}_p : x^r_{pat}=1} \tilde{\lambda}_r$. Then, p and a are chosen such that $\sum_{r \in \hat{\mathcal{R}}_p : x^r_{pat}=1} \tilde{\lambda}_r$ is the most fractional, i.e., $|\sum_{r \in \hat{\mathcal{R}}_p : x^r_{pat}=1} \tilde{\lambda}_r - 0,5|$

is minimized and different from 0,5. Note that this simple branching rule produces search trees with significantly fewer nodes than other branching rules of comparable complexity, such as choosing over all triples (p, a, t), $p \in \{1, \ldots, P\}$, $t \in \{1, \ldots, T\}$, $a \in A_{pt}$, the one that gives the most fractional $\sum_{r \in \hat{\mathcal{R}}_p : x^r_{pat}=1} \tilde{\lambda}_r$, or choosing a triple (p, a, t) randomly. To eliminate a set of fractional solutions which contains $\tilde{\lambda}$, separation occurs and two new nodes are created and added to the search tree. One of them has the additional constraint:

$$\sum_{\substack{r \in \hat{\mathcal{R}}_p: \\ x^r_{pat}=1}} \lambda_r = 0,$$

and the other one has the constraint:

$$\sum_{\substack{r \in \hat{\mathcal{R}}_p: \\ x^r_{pat}=1}} \lambda_r = 1.$$

This branching scheme corresponds to branching on the original variables of the compact formulation CF, and adding branching constraints to the linking constraints [20]. Once a node has been processed and its linear relaxation has been solved by the above column generation scheme, an aggressive pruning strategy is added to the standard pruning process. Given a best known integral solution of value $\bar{z}$, any node is pruned if its lower bound is over the solution of the following (NP-hard) knapsack problem:

$$\begin{aligned} \max \quad & \sum_{p=1}^{P} s_p y_p \\ \text{s.t.} \quad & \sum_{p=1}^{P} s_p y_p < \bar{z} \\ & y_p \in \{0, 1\} \end{aligned}$$

As this knapsack problem has a number of variables equal to P which is generally not large, and the surface areas are generally small, it can be efficiently solved, for example by dynamic programming based algorithms [21].

When looking for a column variable with the minimum reduced cost in the pricing problem of column generation, each dual value corresponding to a branching constraint on an arc $a \in A_{pt}$ is simply subtracted from the cost of

arc a in graph G^p for the shortest path computation. In the next subsection, we present the cutting planes added to accelerate problem solving.

4.3. Cutting planes

During the processing of any node of the search tree, column and row generations are carried out, in that order, in a loop. This loop starts with column generation: a subproblem of $\overline{MP}$, characterized by a set of branching constraints and including a set of inequalities valid for MP, is solved to optimality. Then if the node cannot be pruned, cutting plane generation takes place. When a cut is generated, it is added to the last RMP of the current node, and another iteration of the column-and-row generation loop begins. Otherwise, if no cuts are generated, the algorithm exits from this loop, and either the node is pruned or branching occurs.
Cutting planes are generated as follows. When the last solution given by column generation, noted $\tilde{\lambda}$, is fractional, two separation heuristics try to generate inequalities valid for MP in respect to integrality constraints, but violated by $\tilde{\lambda}$.
Cutting planes are generated from the compact formulation CF. A solution $x^{(\lambda)}$ to $\overline{CF}$ is built from a $\overline{MP}$ solution λ as:

$$x_{pat}^{(\lambda)} = \sum_{\substack{r \in \hat{\mathcal{R}}_p: \\ x_{pat}^r = 1}} \lambda_r \tag{12}$$

After a change of variables $\bar{x}_{pat} = 1 - x_{pat}$, constraints (2) of CF become knapsack constraints:

$$\sum_{p=1}^{P} \sum_{a \in A_{pt}} s_p w_{pac} \bar{x}_{pat} \leq \sum_{p=1}^{P} \sum_{a \in A_{pt}} s_p w_{pac} - d_{ct} \quad \text{for } t = 1, \ldots, T,\ c \in C_t \tag{13}$$

with $\bar{x}_{pat} \in \{0, 1\}$. The two separation heuristics mentioned above try to generate an inequality valid for one of the above knapsack constraints (13), but violated by $x^{(\tilde{\lambda})}$. When cut generation succeeds, the generated inequality valid for the knapsack constraint is uncomplemented and translated in the extended formulation by using variable substitution (12).

The idea of using valid inequalities for the knapsack polytope to tackle more complex 0-1 problems was present in [22], and suggested in the case of a sparse constraint matrix. Note that the sets of variables with non-zero

coefficients in constraints (13) are pairwise disjoint, which results in a sparse constraint matrix. [23] describes several classes of inequalities valid for the knapsack polytope, such as (i) extended cover inequalities and (ii) weight inequalities, which are both generated in our BPC.

(i) Extended cover inequalities are separated by the greedy heuristic of [24] and built as follows. Given a period t and a crop c, we first build a minimal cover D by adding pairs of indices (p, a), with $a \in A_{pt}$, in non-increasing order of the value $\sum_{r \in \hat{\mathcal{R}}_p : x^r_{pat}=1} \tilde{\lambda}_r$, where $\tilde{\lambda}$ is the current fractional solution. We add such indices (p, a) in a greedy way until the sum of their coefficients $s_p w_{pac}$ exceeds the demand d_{ct}. Then we compute $\bar{w} = \max_{(p,a)\in D} w_{pac}$ and set $E = \{(p, a) : w_{pac} \geq \bar{w}\}$. In the case of success, the separation heuristic returns an extended cover inequality of the form:

$$\sum_{(p,a)\in(D\cup E)} \bar{x}_{pat} \leq |D| - 1$$

$$\text{i.e.,} \quad \sum_{(p,a)\in(D\cup E)} x_{pat} \geq |E| + 1$$

which is violated by $x^{(\tilde{\lambda})}$. After variable substitutions (12) we obtain the following inequality valid for MP and violated by $\tilde{\lambda}$:

$$\sum_{(p,a)\in(D\cup E)} \sum_{\substack{r\in\hat{\mathcal{R}}_p: \\ x^r_{pat}=1}} \lambda_r \geq |E| + 1$$

(ii) Weight inequalities are also associated with a period t and a crop $c \in C_t$, and are built in the following way. First, we construct in a greedy way a pack set O by adding in O indices (p, a) in non-increasing order of the value $\sum_{r \in \hat{\mathcal{R}}_p : x^r_{apt}=1} \tilde{\lambda}_r$ as in (i). Instead of adding these indices until demand d_{ct} is covered as in (i), one stops at the last iteration for which the demand remains uncovered and computes the residual capacity:

$$\delta = d_{ct} - \sum_{(p,a)\in O} \sum_{\substack{r\in\hat{\mathcal{R}}_p: \\ x^r_{apt}=1}} w_{pac}\tilde{\lambda}_r$$

We call N the set of indices of variables which are in the knapsack constraints but not in O. The separation heuristic of [23] returns a

weight inequality of the form:

$$\sum_{(p,a)\in O} w_{pac}\bar{x}_{pat} + \sum_{(p,a)\in N} \max\{0, w_{pac} - \delta\}\bar{x}_{pat} \leq \sum_{(p,a)\in O} w_{pac}$$

$$\text{i.e.,} \quad \sum_{(p,a)\in O} w_{pac}x_{pat} + \sum_{(p,a)\in N} \max\{0, w_{pac} - \delta\}x_{pat} \geq \sum_{(p,a)\in N} \max\{0, w_{pac} - \delta\}$$

which is violated by $x^{(\tilde{\lambda})}$. After variable substitutions (12) we get the following inequality valid for MP and violated by $\tilde{\lambda}$:

$$\sum_{(p,a)\in O} \sum_{\substack{r\in\hat{\mathcal{R}}_p:\\ x^r_{apt}=1}} w_{pac}\lambda_r + \sum_{(p,a)\in N} \sum_{\substack{r\in\hat{\mathcal{R}}_p:\\ x^r_{pat}=1}} \max(0, w_{pac}-\delta)\lambda_r \geq \sum_{(p,a)\in N} \max(0, w_{pac}-\delta)$$

A cut generated in the compact formulation CF is therefore translated into a cut for the extended formulation MP according to the mapping (12). Each variable appearing in a cut in the compact formulation has its coefficient modified in the objective function of the pricing problem; however the structure of the pricing problem is unaffected.

4.4. Restricted master heuristic

In the case of a large-size instance, the BPC algorithm may not provide an integer feasible solution in a reasonable amount of time. To increase the efficiency of our BPC, we embedded in it a restricted master heuristic. Restricted master heuristics are the most commonly used primal heuristics with column generation [25]. They consist in solving a restricted master problem (RMP) with integrality constraints on column variables, for example by a branch-and-bound algorithm. Our restricted master heuristic provided very good integer solutions when needed. It is invoked periodically with a time limit, as long as the gap between the worst lower bound and the best upper bound is above a given threshold (if no integer solution is known, the gap is defined as infinite). To build a RMP, a set of columns has to be chosen. Columns r with the highest average value of $\tilde{\lambda}_r$ over the set of closed nodes are selected.

5. Numerical results

We present in tables 1, 2, 3 and 4, a comparison of our Branch-and-Price-and-Cut algorithm (BPC), with an Integer Program Solver for the compact

formulation (IP SOLVER), and a heuristic denoted by HEUR which consists in solving the last restricted master problem obtained by column generation with integrality constraints as a static integer program. All computations reported in this paper have been carried out on a personal computer with an Intel Core i5 processor at 2.50 GHz and 8 Go of RAM. The integer program solver is IBM ILOG CPLEX 12.4 with default settings. BPC has been coded in C++ using COIN-OR BCP 1.3.4.

In the tables, for each approach, we report the value of the best integer solution obtained within a time limit equal to 3600 seconds (*value*) and the number of nodes (*#nodes*) in the search tree. Only in tables 1 and 2, we add a *time* column with the running time in seconds, as most instances are solved to optimality before the time limit of 3600 seconds. When an optimal solution is obtained by one of the two exact methods and is proven optimal, its value is marked by *. Furthermore, we emphasize in bold instances where BPC improves both IP SOLVER and HEUR. In the last row of each table, noted *#best*, we report the ratio of the number of best-found solutions by the number of instances, for each approach. The last column of each table denotes the value of the linear relaxation. This information enables pointing out the hardness of these instances. The larger the gap between the LP value and the best integer value is, the more difficult an instance is since a large gap is longer to close for an exact method.

Each instance, denoted by I_*# crops_# periods_# plots* , is characterized by the number of crops (*# crops*), the number of periods that defines the planning horizon (*# periods*) and the number of plots (*# plots*). The yield of a crop depends on the state of the plot and the transition used to reach that state ; it increases with fallow length l and decreases with cultivation length l'.

In tables 1 and 2, we report numerical results on instances with two crops. The objective is to compare the three approaches on small-sized instances to evaluate the convergence to optimality of the two exact methods.

In table 1, instances have the same characteristics (20 periods and 7 plots). To diversify the sample, the surface area of each plot is drawn in the interval [4,11].
We observe that for all instances, the two exact methods find an optimal solution but BPC is in average 5 times faster than IP SOLVER. Furthermore, the average gap between the LP value and the optimum is equal to 14.4%.

In table 2, instance sizes are progressively increased by varying the number of plots from 7 to 69 with five possible values in that range, while keeping

the average total surface area constant. The number of periods is 20 or 40. For the first of the five pairs (I_2_20_, I_2_40_) the surface area of a plot is uniformly generated in the interval [5,15], for the second in [3.5,10.5], for the third in [2.5, 7.5], for the fourth in [1.5, 4.5] and for the last in [0.5, 1.5]. By reducing plot sizes, we increase the number of available plots. Crop demands of these instances are uniformly drawn in a set of small intervals. In this table, the average gap between the LP value and the best-found integer solution remains large (12.16%).
For that set of instances, BPC outperforms the two other approaches in terms of quality of the best-found integer solution with a *#best* score of 10/10. Furthermore, BPC is in average 25 % faster than IP SOLVER. Also, note that the number of periods of the planning horizon is a critical parameter, with computational times multiplied by a factor up to 10 when passing from 20 to 40 periods. In tables 3 and 4, we present numerical results

instance	BPC			IP SOLVER			HEUR			LP
	#nodes	time	value	#nodes	time	value	#nodes	time	value	value
I_2_20_7 - 1	2881	74.61	22.00*	5598	119.93	22.00*	1	0.19	30.00	18.88
I_2_20_7 - 2	645	24.26	22.80*	33647	455.33	22.80*	1	0.01	22.80	18.71
I_2_20_7 - 3	6093	175.04	24.30*	112702	956.72	24.30*	1	0.24	25.50	18.84
I_2_20_7 - 4	1843	44.45	25.50*	37402	367.38	25.50*	1	0.11	25.50	20.33
I_2_20_7 - 5	7747	159.15	23.00*	42427	479.30	23.00*	1	0.28	30.00	20.34
#best	5 / 5			5 / 5			2 / 5			

Table 1: Computational results for 2 crops, 20 periods and 7 plots

on more difficult instances with a number of plots varying from 30 to 70, a number of crops from 4 to 8 and a number of periods from 6 to 12. For a given triplet *# crops, # periods, # plots*, 5 instances are created, with the same crop demands. The yields and surface areas vary uniformly by a factor between 0.8 and 1.2 from a set of reference values, which enables to create diversified instances. We set $L = 4$ and $L' = 6$ for all instances.
We note that for the last pool of 30 instances, none of the two exact methods converged to optimality within the time limit of 3600 seconds. Tables 3 and 4 confirm the superiority of BPC over the two other approaches in terms of quality of the best-found integer solution with a *#best* score of 33/40 versus 7/40 for the IP solver and 5/40 for the heuristic. Furthermore, we observe that for the I_8_12_70 instances, IP SOLVER failed to find an integer solution.

Instances	BPC			IP SOLVER			HEUR			LP
	#nodes	time	value	#nodes	time	value	#nodes	time	value	value
I_2_20_7 - 1	3	0.58	19.90*	1	0.46	19.90*	1	0.00	20.20	16.69
I_2_40_7 - 1	3	3.39	20.40*	1	1.95	20.40*	1	0.08	29.60	17.64
I_2_20_10 - 1	759	6.73	18.13*	13110	77.80	18.13*	1	0.02	19.81	14.79
I_2_40_10 - 1	3	17.92	19.53*	10538	124.02	19.53*	1	0.12	26.53	15.39
I_2_20_14 - 1	1279	15.51	14.90*	418	4.31	14.90*	1	0.02	17.40	13.43
I_2_40_14 - 1	3385	158.27	16.55*	3652	302.39	16.55*	4282	1.57	18.05	13.74
I_2_20_23 - 1	10699	176.67	13.98*	13868	245.34	13.98*	1	0.04	16.32	12.63
I_2_40_23 - 1	54565	1593.36	**15.81***	49400	3600	16.08	1	0.30	18.57	14.36
I_2_20_69 - 1	200701	3600	**12.40**	64594	3600	12.48	1	0.41	12.76	12.06
I_2_40_69 - 1	52891	3600	**13.31**	4977	3600	13.47	53837	20.42	13.55	12.79
#best	10 / 10			7 / 10			0 / 10			

Table 2: Computational results for 2 crops, 20 to 40 periods and 7 to 69 plots

Also note that the number of crops is a critical parameter for tractability, with a number of nodes often reduced by a factor 10 (within the 3600 seconds time limit) when passing from 4 crops to 8 crops.

Finally, table 2 confirms a natural result which is useful to design sustainable agricultural systems: when the total available land is divided up into a higher number of smaller plots, the set of possible combinations of land plots to cover an identical set of demands is enlarged, therefore the total number of hectares needed for production is reduced and more land is preserved (see [8] for the formal result and proof). Indeed, for $T = 20$, the total surface area needed for covering crop requirements decreases from 19.2 for 7 plots with an average size of 10, to 12.4 for 69 plots with an average size of 1. Hence reducing by a factor 10 the unit size of a plot reduces the total space consumption by 35% with our dataset. For $T = 40$ we obtain similar results, as when the average size of a plot passes from 10 to 1, the total surface area used for production decreases from 20.4 to 13.3. Therefore, the unit size of a plot is a critical parameter for sustainability, regardless of the increased cost of managing and maintaining a larger number of plots.

6. Conclusion

Crop rotation planning problems have received increasing attention from OR researchers in the past five years, especially for sustainable agriculture.

Instances	BPC		IP solver		Heur		LP
	#nodes	value	#nodes	value	#nodes	value	value
I_4_6_30 - 1	781151	19.00	658814	19.00	158996	20.00	18.17
I_4_6_30 - 2	871295	21.20	748944	21.20	130724	21.70	20.51
I_4_6_30 - 3	915997	22.30	864819	22.30	83896	22.80	21.50
I_4_6_30 - 4	803839	18.40	54744	18.40*	1	19.20	17.71
I_4_6_30 - 5	684151	19.60	134818	19.60*	38300	20.60	18.89
I_4_12_30 - 1	199307	**26.00**	102360	26.60	11904400	26.10	24.81
I_4_12_30 - 2	241291	**25.10**	97161	26.20	12946600	26.20	24.00
I_4_12_30 - 3	210301	25.60	83238	25.60	13498647	26.60	24.19
I_4_12_30 - 4	220809	26.40	98142	26.40	13183513	26.40	25.06
I_4_12_30 - 5	214965	**25.70**	89329	26.10	12332147	25.80	24.54
#best (including. ties)	10 / 10		7 / 10		1 / 10		

Table 3: Computational results for 4 crops, 6 or 12 periods and 30 plots

Some papers proposed a column generation approach for covering variants of crop rotation planning, but none has ever proposed neither a branch-and-price nor a branch-and-cut for these problems, to our knowledge. In this paper, a Branch-and-Cut-and-Price (BPC) is proposed for a specific sizing problem introduced by the authors, where space consumption is minimized and seasonal demands are to be covered. This enables sizing with minimum waste the land space needed for production over a large period of time. The BPC proposed in this paper largely outperforms the direct solving of a compact formulation of the problem, as well as a heuristic that consists in solving the last restricted master program with integer column variables. Numerical results are robust over a set of 55 instances where one varies the number of crops, the number of periods and the number of available plots or unit sizes of a plot. The effectiveness of BPC is evidenced despite the fact that the pricing subproblem of column generation has the integrality property, and thus the LP bound of the extended formulation is not better but equal to that of the compact formulation. Another learning point of the paper is to measure how much dividing up the land space into smaller plots can reduce the total number of hectares needed for production, up to 35% when the unit size of a plot is divided by 10 for the same demands to cover.

Instances	BPC		Compact formulation		CG+IP		LP
	#nodes	value	#nodes	value	#nodes	value	value
I_4_8_70 - 1	369295	39.80	127887	39.80	14490500	40.00	39.13
I_4_8_70 - 2	379093	39.90	161983	39.80	14050751	40.10	39.09
I_4_8_70 - 3	408209	**39.40**	114737	39.50	1730322	39.50	38.79
I_4_8_70 - 4	411009	39.20	122798	39.10	15012516	39.60	38.56
I_4_8_70 - 5	418393	41.60	157534	41.60	11974769	41.90	40.89
I_4_12_70 - 1	117843	49.30	22989	48.90	12324754	49.30	47.87
I_4_12_70 - 2	113611	**47.40**	17687	47.60	11808027	47.50	46.29
I_4_12_70 - 3	134575	**48.80**	23173	48.90	10568251	49.10	47.69
I_4_12_70 - 4	113713	48.70	14096	48.30	9941989	48.70	47.45
I_4_12_70 - 5	136567	**47.40**	23455	47.50	11126907	47.60	46.41
I_6_8_70 - 1	111647	**35.60**	31439	35.70	10043917	36.00	34.33
I_6_8_70 - 2	109529	**39.00**	39770	39.20	9495680	39.40	37.70
I_6_8_70 - 3	114431	**35.70**	42528	36.10	10135149	35.90	34.55
I_6_8_70 - 4	101635	36.60	33354	36.60	9888217	36.70	35.32
I_6_8_70 - 5	108503	**36.70**	31903	36.80	9327708	37.40	35.32
I_6_12_70 - 1	24987	**43.70**	1429	45.40	5818061	43.90	41.79
I_6_12_70 - 2	29971	**43.10**	1319	47.30	5661629	43.30	41.38
I_6_12_70 - 3	29969	**45.00**	1250	46.40	6382705	45.90	42.72
I_6_12_70 - 4	25961	44.20	973	48.30	5396047	44.20	42.24
I_6_12_70 - 5	30979	**44.70**	1304	45.20	6473536	44.90	42.46
I_8_8_70 - 1	22969	**49.60**	3657	49.70	6730924	50.20	47.55
I_8_8_70 - 2	20935	55.30	2389	55.60	7397200	55.20	52.60
I_8_8_70 - 3	22933	**51.50**	4890	51.90	6845598	52.20	49.04
I_8_8_70 - 4	25959	**52.20**	5090	52.70	6800695	53.00	50.38
I_8_8_70 - 5	22951	**54.00**	3950	54.60	6902827	54.70	51.76
I_8_12_70 - 1	4991	60.10	1	-	3869550	59.70	56.13
I_8_12_70 - 2	4993	**61.10**	1	-	4214440	61.50	57.51
I_8_12_70 - 3	4991	60.60	1	-	3965560	60.00	56.34
I_8_12_70 - 4	4991	**61.90**	1	-	4058619	62.40	58.30
I_8_12_70 - 5	4993	**60.90**	1	-	4193600	61.60	56.95
#best (including. ties)	23 / 30		7 / 30		4 / 30		

Table 4: Computational results for 4, 6 and 8 crops with 8 or 12 periods and 70 plots

References

[1] M. A. Altieri, et al., Agroecology: the science of sustainable agriculture., no. 2. ed., Westview Press, 1995.

[2] S. R. Gliessman, Agroecology: ecological process in sustainable agriculture, Ann Arbor Press, Michigan.

[3] D. Tilman, K. G. Cassman, P. A. Matson, R. Naylor, S. Polasky, Agricultural sustainability and intensive production practices, Nature 418 (6898) (2002) 671–677.

[4] N. K. Detlefsen, A. L. Jensen, Modelling optimal crop sequences using network flows, Agricultural Systems 94 (2) (2007) 566–572.

[5] T. El-Nazer, B. A. McCarl, The choice of crop rotation: A modeling approach and case study, American Journal of Agricultural Economics 68 (1) (1986) 127–136.

[6] W. Haneveld, A. W. Stegeman, Crop succession requirements in agricultural production planning, European Journal of Operational Research 166 (2) (2005) 406–429.

[7] L. M. R. dos Santos, P. Michelon, M. N. Arenales, R. H. S. Santos, Crop rotation scheduling with adjacency constraints, Annals of Operations Research 190 (1) (2011) 165–180.

[8] L. Alfandari, J. Lemalade, A. Nagih, G. Plateau, A mip flow model for crop-rotation planning in a context of forest sustainable development, Annals of operations research 190 (1) (2011) 149–164.

[9] L. M. R. dos Santos, A. M. Costa, M. N. Arenales, R. H. S. Santos, Sustainable vegetable crop supply problem, European Journal of Operational Research 204 (3) (2010) 639–647.

[10] A. M. Costa, L. M. R. dos Santos, D. J. Alem, R. H. Santos, Sustainable vegetable crop supply problem with perishable stocks, Annals of Operations Research (2013) 1–19.

[11] M. Rădulescu, C. Z. Rădulescu, G. Zbăganu, A portfolio theory approach to crop planning under environmental constraints, Annals of Operations Research (2011) 1–22.

[12] J. Dury, N. Schaller, F. Garcia, A. Reynaud, J. E. Bergez, Models to support cropping plan and crop rotation decisions. a review, Agronomy for sustainable development 32 (2) (2012) 567–580.

[13] G. B. Dantzig, P. Wolfe, Decomposition principle for linear programs, Operations research 8 (1) (1960) 101–111.

[14] J. Sadki, Problèmes de couverture en nombres entiers : génération de colonnes, heuristiques d'approximation garantie et schémas hybrides. applications en transport ferroviaire et en planification de production, Ph.D. thesis, Université Paris 13 (2011).

[15] M. R. Garey, D. S. Johnson, "strong"np-completeness results: Motivation, examples, and implications, Journal of the ACM (JACM) 25 (3) (1978) 499–508.

[16] M. E. Lübbecke, J. Desrosiers, Selected topics in column generation, Operations Research 53 (6) (2005) 1007–1023.

[17] C. Lemaréchal, The omnipresence of lagrange, Annals of Operations Research 153 (1) (2007) 9–27.

[18] C. Barnhart, E. L. Johnson, G. L. Nemhauser, M. W. Savelsbergh, P. H. Vance, Branch-and-price: Column generation for solving huge integer programs, Operations research 46 (3) (1998) 316–329.

[19] F. Vanderbeck, On dantzig-wolfe decomposition in integer programming and ways to perform branching in a branch-and-price algorithm, Operations Research 48 (1) (2000) 111–128.

[20] G. Gamrath, Generic branch-cut-and-price, Ph.D. thesis, Technische Universitt Berlin (2010).

[21] H. Kellerer, U. Pferschy, D. Pisinger, Knapsack problems, Springer, 2004.

[22] H. Crowder, E. L. Johnson, M. Padberg, Solving large-scale zero-one linear programming problems, Operations Research 31 (5) (1983) 803–834.

[23] K. Kaparis, A. N. Letchford, Separation algorithms for 0-1 knapsack polytopes, Mathematical programming 124 (1-2) (2010) 69–91.

[24] Z. Gu, G. L. Nemhauser, M. W. Savelsbergh, Lifted cover inequalities for 0-1 integer programs: Computation, INFORMS Journal on Computing 10 (4) (1998) 427–437.

[25] C. Joncour, S. Michel, R. Sadykov, D. Sverdlov, F. Vanderbeck, Column generation based primal heuristics, Electronic Notes in Discrete Mathematics 36 (2010) 695–702.

ESSEC Business School
Avenue Bernard Hirsch
BP 50105
95021 Cergy-Pontoise Cedex
France

Tél. +33 (0)1 34 43 30 00
Fax +33 (0)1 34 43 30 01

www.essec.fr

ESSEC Executive Education
CNIT BP 230
92053 Paris-La Défense
France

Tél. +33 (0)1 46 92 49 00
Fax +33 (0)1 46 92 49 90

http://formation.essec.fr

ESSEC Business School
Singapore Campus
100 Victoria Street
National Library Building # 13-02
Singapore 188064

essecasia@essec.fr
Tél. +65 6884 9780
Fax +65 6884 9781

www.essec.edu

Informations

Alison Bougi
+33 (0)1 34 43 33 58
bougi@essec.fr
www.essec.fr
research.center@essec.fr

ISSN 1291-9616

ESSEC BUSINESS SCHOOL