# A graph-based, exact solution to a Pareto optimization problem under partial order constraints

Philippe Nghe

*FOM Institute AMOLF, Science Park 104,*

*1098XG Amsterdam, the Netherlands and*

*Laboratoire de Biochimie, UMR CNRS-ESPCI Chimie Biologie Innovation 8231,*

*ESPCI, 10 rue Vauquelin, 75005 Paris, France*\*

Bela M. Mulder, Sander J. Tans

*FOM Institute AMOLF, Science Park 104,*

*1098XG Amsterdam, the Netherlands*

(Dated: July 19, 2016)

## Abstract

We developed a graph-based method to solve the multi-objective optimization problem of minimizing or maximizing subsets of $N$ bounded variables under partial order constraints. This problem, motivated by the optimization of the response of multi-input monotone systems applied to biological gene networks, can find applications in other contexts such as task scheduling. We introduce a mapping between coloured graphs (Hasse diagrams) and polytopes associated with an ideal point, and find an exact closed-form description of the Pareto optimal set using a dynamic program based on edge contractions. The proof of the algorithm is based on decomposition properties of Pareto optimal sets that follow from elementary set operations, notably a maximality property valid for compact ensembles. In the general case, the Pareto front is found in $O(N.N_P)$ steps, where $N$ is the number of variables and $N_P$ the number of faces of the Pareto front. Using a parameterized complexity approach, the computation and the representation of the solution reaches $O(N)$ for series-parallel graphs when the size of monochromatic connected components is bounded.

## INTRODUCTION

Pareto optimality is a natural extension of the concept of maximum to multi-objective optimization problems. A solution is part of the Pareto optimal set, or Pareto front, if it is impossible to improve one objective without worsening another. Instead of imposing an arbitrary aggregation of the different objectives into a single scalar function, Pareto optimality keeps track of all potentially interesting solutions in the presence of trade-offs. The Pareto approach, originally introduced in economics [13, 18], has shown to be useful in many engineering applications [7, 8, 20], decision-making analysis [19], more recently in biology [14, 15].

Here we apply the Pareto approach to the optimization of the response of multi-input monotone systems, which are widely used to describe input-output systems in control theory [2]. Consider a system which processes a multi-input vector $I \in \mathbb{R}^k$ into a single output $O = f(I) \in \mathbb{R}$, each coordinate of $f$ being monotone. Consider now a list of such input vectors $I_1, ..., I_N$. The natural order on $\mathbb{R}$ induces a partial order between the elements $I_i \in \mathbb{R}^k$, which in turn, due to the monotonicity of $f$, induces partial order constraints between the values $O_1, ..., O_N$. These constraints limit the space accessible to the vector $(O_1, ..., O_N)$. We then introduce an optimization problem: certain values of $O_i$ have to be maximized while other have to be minimized. The question is to know how far the function $f$ can be optimized to fullfill this multi-objective problem, while keeping its monotonicity properties unaffected.

This problem can be formulated beyond the framework of monotone systems, as $f$ and the $I_i$ can be essentially seen as a source of partial order constraints. The most general formulation of our problem is then to start directly from partial order constraints between bounded values $O_1, ..., O_N$, with either a minimization or a maximization objective associated to each of the $O_i$, and search for the Pareto optimal solutions of this problem. Geometrically, setting the bounded $O_i$ between 0 and 1, the space compatible with the constraints is a convex polytope in the hypercube $[0, 1]^N$, and we look for the Pareto front of this polytope given an ideal point located on a corner of the hypercube.

The monotone systems formalism applies in particular to the evolution of gene regulatory networks in biology, which is the initial motivation of the present authors [17]. In this context: each $I_i$ represents a combination of environmental signals; $f$ is the signal processing

performed by the gene regulatory network, where mutations impact the strength of the interactions but not their sign [12]; the $O_j$ are the output of the network (ex: the expression of a specific gene), which can be either beneficial or detrimental, setting selective pressure toward respectively either maximization or minimization.

The broader formulation of the problem can find applications in fields more specific to operations research, such as task scheduling: some tasks have to be realized in a certain temporal order relative to each other due to design constraints (e.g. the assembly of the different parts of a car) [11], with some having to be realized as soon as possible and others as late as possible due to externalities (e.g. stock management, supply constraints, or processing unit occupancy).

In section 2, we demonstrate general properties of Pareto sets in relation with set operations, introduce the concept of Pareto self-dominance. We demonstrate a number of properties to break multi-objective optimization problems into simpler sub-problems, useful for our later demonstration, notably a maximality property valid for compact set (Proposition 3). In section 3, we provide a graph formulation of our multi-objective optimization under partial order constraints problem, using a Hasse diagram [16], which we colour according to optimization objectives. We then describe and prove a dynamic programming algorithm based on successive edge contractions with appropriate vertex colouring rules. In section 4 we discuss the complexity of our algorithm as a function of the number of variables $N$ and to the complexity of the Pareto front $N_P$. We propose an improved version of the algorithm in $O(N.N_P)$. We finally discuss how a parameterized complexity approach [1] can provide a combinatorial description of the Pareto front, with a complexity of $O(N)$ in the case of series-parallel partial orders with monochromatic connected components of bounded size.

## DEFINITIONS AND SOME PROPERTIES OF PARETO OPTIMALITY

We consider a partially ordered set $(\Omega, \succeq)$ and define the corresponding strict order $x \succ y \iff (x \succeq y$ and $x \neq y)$. For $x \in \Omega$, we denote the *upper contour set* and the *strictly upper contour set* respectively by $\succeq(x) = \{y \in \Omega \text{ such that } y \succeq x\}$ and $\succ(x) = \{y \in \Omega \text{ such that } y \succ x\}$.

The (strong) Pareto optimal set of $X \subseteq \Omega$, denoted by $Par(X)$, contains the elements of

$X$ which are not stricly dominated by another element of $X$:

$$Par(X) = \{x \in X \text{ such that } \succ(x) \cap X = \varnothing\}. \tag{1}$$

Equivalently we have

$$\forall x \in X, x \in Par(X) \iff \forall y \in X, y \succeq x \implies y = x. \tag{2}$$

The following propositions on Pareto optimal sets will be of use in the proof of our algorithm in Section .

**Proposition 1** *Let $A$ and $B$ be subseteqs of $\mathbb{R}^N$. Then $Par(A) \cap B \subseteq Par(A \cap B)$.*

*Proof:* Consider $x \in Par(A) \cap B$.
$\succ(x) \cap (A \cap B) \subseteq \succ(x) \cap A = \varnothing$ because $x \in Par(A)$.
As $x \in A \cap B$, by definition of Pareto optimality $x \in Par(A \cap B)$.

**Proposition 2** *Consider $A_1, ..., A_I \subseteq \Omega$. $Par\left(\bigcup_{i=1}^I A_i\right) \subseteq \bigcup_{i=1}^I Par(A_i)$.*

*Proof:* Consider $x \in Par\left(\bigcup_{i=1}^I A_i\right)$. We have

$$\varnothing = \succ(x) \cap \bigcup_{i=1}^I A_i = \bigcup_{i=1}^I \succ(x) \cap A_i$$

This implies $\forall i, \succ(x) \cap A_i = \varnothing$. Given that $x \in \bigcup_{i=1}^I A_i$, $\exists i_0$ such that $x \in A_{i_0}$. Considering that in particular $\succ(x) \cap A_{i_0} = \varnothing$, we have $x \in Par(A_{i_0}) \subseteq \bigcup_{i=1}^I Par(A_i)$.

**Definition 1** *We say that a set $A \subseteq \Omega$ is self-dominated if $\forall x \in A, \exists y \in Par(A)$ such that $y \succeq x$.*

We will now prove a generalization of the following property of the maximum in $\mathbb{R}$ to Pareto optimality in $\mathbb{R}^N$ : take two closed segments $A$ and $B$ in $\mathbb{R}$, such that $A \subseteq B$ and $max(A) \geq max(B)$. This imposes $max(A) = max(B)$.

**Proposition 3 (maximality property for self-dominated sets)** *Consider $A, B \subseteq \mathbb{R}^N$, $B$ self-dominated. If $A \subseteq B$ and $Par(B) \subseteq Par(A)$ then $Par(A) = Par(B)$.*

*Proof:* We only need to show $Par(A) \subseteq Par(B)$. For that consider $x \in Par(A)$. $x \in A \subseteq B$ by hypothesis. By Definition 1,

$$\exists y \in Par(B) \text{ such that } y \succeq x$$

As $Par(B) \subseteq Par(A)$, in particular $y \in Par(A) \subseteq A$. By relation (2), $x = y \in Par(B)$.

**Proposition 4** *Consider $A \subseteq \Omega$ self-dominated. Suppose $x \in B$ is a necessary condition for $x \in Par(A)$, then $Par(A) = Par(A \cap B)$.*

*Proof:* The hypothesis $Par(A) \subseteq B$ implies $Par(A) \subseteq Par(A) \cap B \subseteq Par(A \cap B)$ by Proposition 1. The result comes from Proposition 3 replacing $A$ and $B$ by respectively $A \cap B$ and $A$.

**Proposition 5** *Consider $X \subseteq \Omega$ self-dominated. Suppose there exist $A_1, ..., A_I \subseteq X$ such that $Par(X) = Par(\bigcup_{i=1}^{I} A_i)$. Then $Par(X) = Par\left(\bigcup_{i=1}^{I} Par(A_i)\right).$*

*Proof:* We just need to apply Proposition 3 to $A = X$ and $B = \bigcup_{i=1}^{I} Par(A_i)$, the initial inclusion hypothesis being given by Proposition 2.

**Remark–** Proposition 5 is in particular true when $X = \bigcup_{i=1}^{I} A_i$ and indicates that $Par(\bigcup_{i=1}^{I} A_i) = Par\left(\bigcup_{i=1}^{I} Par(A_i)\right).$

**Proposition 6** *Suppose that in addition to the hypotheses of Proposition 5, $A_1$ is self-dominated and verifies $\exists B_1, ..., B_J \subseteq A_1$ such that $Par(A_1) = Par(\bigcup_{j=1}^{J} B_j)$. We then have*

$$Par(X) = Par\left[\bigcup_{j=1}^{J} Par(B_j) \cup \bigcup_{i=2}^{I} Par(A_i)\right]$$

.

*Proof:* Applying successively Proposition 5 to $A_1$ instead of $X$, the remark of Proposition 5 in the backward direction, then in the forward direction and finally using the fact that

5

$Par(Par(Y)) = Par(Y)$ for any set $Y \subseteq \Omega$,

$$Par(X) = Par \left[ \bigcup_{i=1}^{I} Par(A_i) \right]$$

$$= Par \left[ Par \left( \bigcup_{j=1}^{J} Par(B_j) \right) \cup \bigcup_{i=2}^{I} Par(A_i) \right]$$

$$= Par \left[ \bigcup_{j=1}^{J} Par(B_j) \cup \bigcup_{i=2}^{I} A_i \right]$$

$$= Par \left[ \bigcup_{j=1}^{J} Par(Par(B_j)) \cup \bigcup_{i=2}^{I} Par(A_i) \right]$$

$$= Par \left[ \bigcup_{j=1}^{J} Par(B_j) \cup \bigcup_{i=2}^{I} Par(A_i) \right]$$

Finally, we note that if $A$ is a closed, non-empty subseteq of $\mathbb{R}^N$ which has an upper bound in the sense of the Pareto order (*i.e.* $\exists x_M \in \mathbb{R}^N$ such that $\forall x \in A, x_M \succeq x$), then $Par(A) \neq \varnothing$ [4]. As a consequence $A$ is self-dominated, as we show below.

**Proposition 7** *When $A \subseteq \mathbb{R}^N$ closed, non-empty and Pareto bounded from above then $A$ is self-dominated.*

*Proof:* Consider $x \in A$. $\succeq (x) \cap A$ is closed, non-empty and Pareto bounded from above. Thus $Par (\succeq (x) \cap A) \neq \varnothing$ and we can consider $y \in Par (\succeq (x) \cap A)$, so that $y \succeq x$ and $y \in A$. Now, as $\succ (y) \subseteq \succeq (x)$

$$\succ (y) \cap A = (\succ (y) \cap \succeq (x)) \cap A =\succ (y) \cap (\succeq (x) \cap A) = \varnothing$$

which proves $y \in Par (A)$.

Proposition 7 ensures that we can use Propositions 1, 2, 3 and 4 for the proof of the algorithm in the context of $[0, 1]^N$ presented in the next part. Propositions 5 and 6 will not be used for the demonstration of our specific algorithm, but to discuss the complexity of more naive algorithms in section . Indeed, they can be used generally in multi-objective optimization problems for reducing the Pareto set search into sub-problems. Proposition 5 indicates that the Pareto front can be found by taking the Pareto front of the assembled Pareto fronts of any decomposition of the search space. Proposition 6 extends this result to any finite number of recursive splittings of the search space, and shows that it is necessary to search only one time the Pareto set of the assembled Pareto sets of the terminal sub-problems.

# PARETO OPTIMIZATION UNDER PARTIAL ORDER CONSTRAINTS

## Formulation of the problem and preliminary definitions

***Pareto order*** — We consider $N$ bounded variables $(x_1, ..., x_N) \in \Omega = [0, 1]^N$ and introduce for convenience the notation $x_0 = 0$ and $x_{N+1} = 1$. Our multi-objective optimization consist of minimizing some of the variables and maximizing all the others. Formally, we have a partition $\{\mathcal{N}_+, \mathcal{N}_-\}$ of the index set $\mathcal{N} = \{1, 2, \ldots, N\}$. We define a Pareto ordering $\succeq$ on $\mathbb{R}^N$ with the *signature* $\{\mathcal{N}_+, \mathcal{N}_-\}$ as follows: if $x = (x_1, x_2, \ldots, x_N) \in \mathbb{R}^N$ and $y = (y_1, y_2, \ldots, y_N) \in \mathbb{R}^N$ then

$$x \succeq y \iff \forall i \in \mathcal{N}_+ : x_i \geq y_i \text{ and } \forall j \in \mathcal{N}_- : x_j \leq y_j, \tag{3}$$

i.e. for the variables in the *ascending* set $\{x_i | i \in \mathcal{N}_+\}$ "larger is better", while for those in the *descending* set $\{x_j | j \in \mathcal{N}_-\}$ "smaller is better". In other terms, the ideal point has coordinates 0 for indexes in $\mathcal{N}_-$ and 1 for indexes in $\mathcal{N}_+$. We define the corresponding (weak) strict Pareto order through $x \succ y \iff x \succeq y \wedge x \neq y$. Partial order constraints correspond to a set of (weak) inequalities of the type $x_i \geq x_j$ between coordinates of $x \in \Omega$.

***Vertex colouring*** — We introduce a graph stucture which vertices $V_I$ are associated with a group of variables $\{x_i\}_{i \in I}, I \subseteq \{0, ..., N+1\}$, called *aggregate*, and noted $x_I$. An aggregate $x_I$ represents a state such that $\forall i, j \in I, x_i = x_j$, a value noted $x_I$. A vertex can be of one of the four following natures (or colour indicated in parenthesis as used in the figures):

- if $I \subseteq \mathcal{N}_-$, $V_I$ is a *descending vertex* (red);

- if $I \subseteq \mathcal{N}_+$, $V_I$ is an *ascending vertex* (blue);

- if $I$ comprises indexes from both $\mathcal{N}_-$ and $\mathcal{N}_+$, $V_I$ is a *Trade-Off Vertex* or *TOV* (grey);

- if $0 \in I$ or $N + 1 \in I$, $V_I$ is a *boundary vertex* (black).

***Partial order constraints*** — Edges express order constraints between variables, according to the convention of so-called Hasse diagrams: edge $E(i \to j)$ points from $V_i$ to $V_j$ if and only if $x_i \geq x_j$, where $x_i$ (resp. $x_j$). The inequality constraints can be consistently carried over to aggregates provided their index-sets $I$ and $J$ are disjoint, and we will use the obvious notation $x_I \geq x_J$. Variables engaged in cyclical inequalities are trivially equal to each other and thus are assumed to be aggregated, resulting in acyclic diagrams.

***Relations between vertices*** — Two vertices $V_i$ and $V_j$ connected by $E(i \to j)$ are said to be *conflicting* if $V_i$ is descending and $V_j$ is ascending. We say that a vertex $V_i$ *aims for* another vertex $V_j$ if: either $V_i$ is descending and points to $V_j$ via an edge $E(i \to j)$; or the other way round, $V_i$ is ascending and is pointed from $V_j$ via an edge $E(j \to i)$, independently of the colour of $V_j$. For example, conflicting vertices necessarily aim for each other. A maximal connected component of the Hasse diagram exclusively comprising ascending or descending vertices is called a *monotone connected components*. A vertex qualified as *extremal* (in the sense of the monotone connected components) if it only aims for vertices of different colour. Note that conflicting vertices are not necessarily extremal as they may point to other vertices of the same nature, and extremal vertices are not necessarily conflicting as they may point to TOVs or to boundary vertices.

***Edge contractions rules*** — An edge contraction between two vertices $V_I$ and $V_J$ consist of removing $E(I \to J)$, and replacing $V_I$ and $V_J$ by a unique vertex $V_K$, where $K = I \cup J$. While all vertices that are not affected by the contraction are of constant colour, the colour of $V_K$ is determined consistently with the colours definitions:

- if $V_I$ and $V_J$ are ascending, $V_K$ is ascending;

- if $V_I$ and $V_J$ are descending, $V_K$ is descending;

- if $V_I$ or $V_J$ is a boundary vertex, $V_K$ is a boundary vertex;

- if $V_I$ and $V_J$ are different and none of them is a boundary vertex, $V_K$ is a TOV;

***Resolution tree*** — We have just introduced all the notions for representing our initial problem and any state resulting from edge contractions in the form of a 4 colours directed acyclic graphs, for which we reserve the term *Hasse diagrams*, or *diagrams*, noted $\mathcal{H}$ with an index. The stages of the dynamic algorithm described below generate another sort of graph, which we call the *resolution tree*. To avoid any confusion between the two sorts of graphs, the vertices of the resolution tree are called *nodes*. Each node of the resolution tree corresponds to a Hasse diagram, and each edge of the resolution tree corresponds to an operation of edge contraction applied to a Hasse diagram. A *branch* will only refer to a branch of the resolution tree. A vertex aiming for several other vertices in $\mathcal{H}$ will be called a *junction*, whereas a node connected to several downstream nodes in the resolution tree will be called a *branching*.

***Re-formulation of the problem*** — We finally call $\Omega_{\mathcal{H}}$ the subspace of $\Omega$ which verifies the partial order relations represent by the Hasse diagram $\mathcal{H}$. With this representation, our main goal can be formulated in the following way:

*Let $\Omega_{\mathcal{H}} \subseteq [0,1]^N$ be the space of all vectors respecting the partial order constraints represented by $\mathcal{H}$. Determine $Par(\Omega_{\mathcal{H}})$, the set of vectors optimal under the Pareto order $\succeq$ on $\mathbb{R}^N$.*

**Statement of the algorithm**

The algorithm starts by setting $\mathcal{H}_0 = \mathcal{H}$. Steps 1 to 4 described below (and illustrated in Figure 1) are then recursively applied to all diagrams $\mathcal{H}_{n,t}$, $t = 1, ..., T_n$, generated at depth $n$ until step 2 can no longer be performed, *i.e.* the diagram in question no longer contains any ascending or descending vertex (eqivalently, only contains TOVs and boundary vertices):

1. Perform a *transitive reduction* of $\mathcal{H}_n$, *i.e.* remove any direct edge $E(u \to v)$ if there exists a longer *path* from $V_u$ to $V_v$ on $\mathcal{H}_n$.

2. Select an extremal vertex $V_i$.

3. Consider the vertices $V_k, k = 1, ..., K$ which $V_i$ aims for. There is always at least one such vertex, in limiting cases provided by boundary vertices. Define the diagrams $\mathcal{H}_{n,k}, k = 1, ..., K$ by respectively contracting the edge connecting $V_i$ and $V_k$ according to the colouring rules defined in section .

At the end of this branching process, we are left with a collection of terminal graphs $\mathcal{H}_t, t = 1, ..., T$, and we posit that the solution of the initial problem is:

$$Par(\Omega_{\mathcal{H}}) = \bigcup_{t=1}^{T} \Omega_{\mathcal{H}_t}.$$

**Proof of the algorithm**

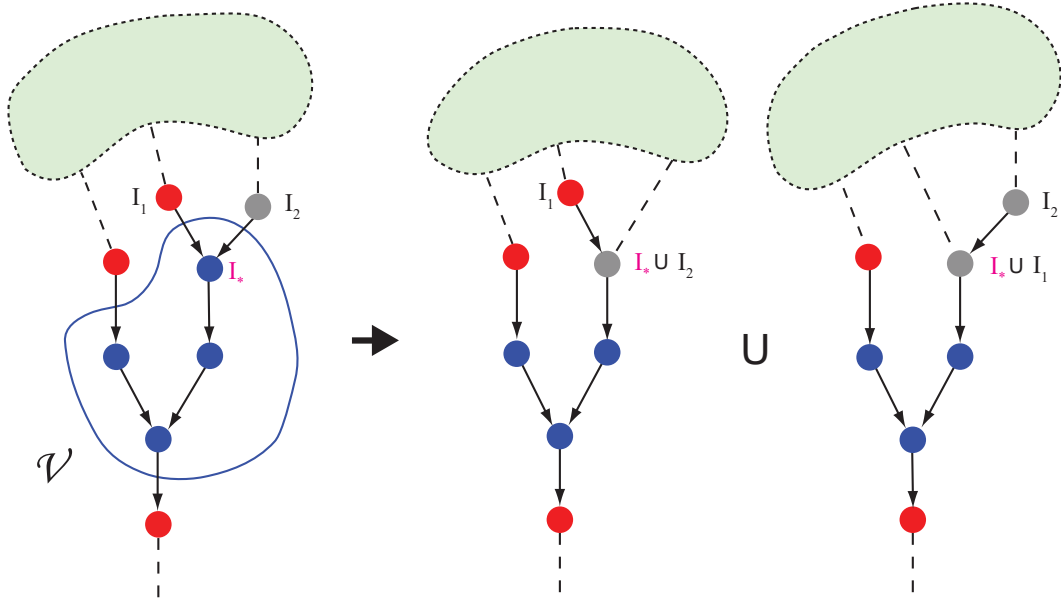Our proof of the algorithm described above proceeds in three steps:

Figure 1: Graphical illustration of the key steps in our algorithm: Right: selection of an ascending connected component $\mathcal{V}$ and identification of an extremal vertex $V_{I_*}$. Left: Aggregation of $V_{I_*}$ with two conflicting vertices $V_{I_1}$ and $V_{I_2}$, yielding a branching of the resolution tree into two diagrams, the union of which solutions results in the Pareto front of the parent diagram. Color code: filled blue circle: ascending vertex, filled red circle: descending vertex, filled grey circle: TOV.

**Step 1** We first show that for each iteration of the algorithm, $Par(\Omega_{\mathcal{H}_n}) = Par(\bigcup_{k=1}^{K} \Omega_{\mathcal{H}_{n,k}})$.

**Step 2** Next we show that $Par(\bigcup_{k=1}^{K} \Omega_{\mathcal{H}_{n,k}}) = \bigcup_{k=1}^{K} Par(\Omega_{\mathcal{H}_{n,k}})$.

**Step 3** Finally we show that the terminal graphs satisfy $Par(\Omega_{\mathcal{H}_t}) = \Omega_{\mathcal{H}_t}$.

For brevity's sake we treat only the case of an ascending connected component, the demonstration being easily adapted for a descending connected component.

### Step 1

Consider an ascending subgraph $\mathcal{V}$ in $\mathcal{H}_n$ and a maximal vertex $V(x_{I_*}) \in \mathcal{V}$. There are two possibilities:

1. The vertex $V_{N+1}$ representing the upper bound is directly connected to $V_{I_*}$. In this case $V_{N+1}$ is the unique vertex pointing to $V_{I_*}$: if there would be another vertex $V$ pointing to $V_{I_*}$, there would exist a chain of vertices pointing from $V_{N+1}$ to $V$, as $V_{N+1}$ is the global upper bound, which would contradict the fact that we have taken the transitive reduction of the diagram.

2. One or more vertices $V_{I_k}, k = 1, ..., K$ point to $V(x_{I_*})$, none of which has constant value.

We then define the sets

$$\omega\left(I_* | I_k\right) = \left\{x \in \mathbb{R}^N \text{ such that } x_{I_*} = x_{I_k}\right\}.$$

A necessary condition for $x \in Par(\Omega_{\mathcal{H}_n})$ is $x \in \bigcup_{k=1}^{K} \omega\left(I_* | I_k\right)$. Suppose otherwise that $\forall k : x_{I_*} < x_{I_k}$ (or in the case 1 above $x_{I_*} < x_0$). Then there exists $\epsilon > 0$ such that for all $k : x_{I_*} + \epsilon < x_{I_k}$. If we denote by $\epsilon_{I_*} \in \mathbb{R}^N$ the vector with coordinates $x_j = \epsilon$ for $j \in I_*$ and otherwise $0$, we have for all $x + \epsilon_{I_*} \in \Omega_{\mathcal{H}_n}$ and $x + \epsilon_{I_*} \succ x$ as by assumption $I_* \subseteq \mathcal{N}_+$. This contradicts $x \in Par(\Omega_{\mathcal{H}_n})$.

As by definition, $\Omega_{\mathcal{H}_n} \cap \omega\left(I_* | I_k\right) = \Omega_{\mathcal{H}_{n,k}}$, we can use Proposition 4 and have that $Par(\Omega_{\mathcal{H}_n}) = Par(\Omega_{\mathcal{H}_n} \cap \bigcup_{k=1}^{K} \omega\left(I_* | I_k\right)) = Par(\bigcup_{k=1}^{K} \Omega_{\mathcal{H}_{n,k}})$.

### Step 2

We only have to show that $Par(\bigcup_{k=1}^{K} \Omega_{\mathcal{H}_{n,k}}) \supset \bigcup_{k=1}^{K} Par(\Omega_{\mathcal{H}_{n,k}})$, the inclusion in the other direction directly following from Proposition 2. We only discuss the case $k \geq 2$ as the result is trivial otherwise (in particular in case 1 of the first step).

11

Consider two distinct indices from the set $\{1, .., K\}$, which we can take to be 1 and 2 without loss of generality. Consider $x \in \Omega_{\mathcal{H}_{n,1}}$ and suppose there exists $y \in \Omega_{\mathcal{H}_{n,2}}$ such that $y \succeq x$. Then by virtue of $V_{I_*}$ being an ascending vertex and by definition of the Pareto order:

$$y_{I_*} \geq x_{I_*} \tag{4}$$

Next, as $V_{I_*}$ was maximal within its ascending subgraph, any vertex pointing to it must contain at least one descending variable, from which follows that

$$x_{I_1} \geq y_{I_1}. \tag{5}$$

Indeed, the latter inequality is trivially true if all $i \in I_1$ label descending variables, i.e. $I_1 \subseteq \mathcal{N}_-$. If not, we can choose an $i_+ \in I_1 \cap \mathcal{N}_+$ and $i_- \in I_1 \cap \mathcal{N}_-$ for which the statement $y \succeq x$ implies both $y_{i_+} \geq x_{i_+}$ and $x_{i_-} \geq y_{i_-}$ and hence, together with $x_{i_+} = x_{i_-}$ and $y_{i_+} = y_{i_-}$ by definition of the aggregates, we obtain the equality $x_{I_1} = y_{I_1}$.

We also have $x \in \Omega_{\mathcal{H}_{n,1}} \subseteq \omega(I_*|I_1) = \{z \in \mathbb{R}^N \text{ such that } z_{I_*} = z_{I_1}\}$ implying:

$$x_{I_*} = x_{I_1}. \tag{6}$$

Finally, $y \in \Omega_{\mathcal{H}_{n,2}} \subseteq \Omega_{\mathcal{H}_n}$ and in $\mathcal{H}_n$, vertex $V_{I_1}$ points to $V_{I_*}$ by hypothesis, which implies:

$$y_{I_1} \geq y_{I_*}. \tag{7}$$

Examining the above relations in the order (5-7-4-6), we see that all the variables at play must be equal, in particular $y_{I_1} = y_{I_*}$. Then $y \in \omega(I_*|I_1)$ and consequently $y \in \Omega_{\mathcal{H}_{n,1}}$. To summarize, we have just demonstrated:

$$\forall j \in \{1, ..., K\}, \forall x \in \Omega_{\mathcal{H}_{n,j}} : (y \in \bigcup_{k=1}^{K} \Omega_{\mathcal{H}_{n,k}} \text{ and } y \succeq x) \Rightarrow (y \in \Omega_{\mathcal{H}_{n,j}}). \tag{8}$$

Now, if we take in particular $x \in Par(\Omega_{\mathcal{H}_{n,j}})$ in relation (8), relation (2) implies that $y = x$ by maximality of $x$ in $\Omega_{\mathcal{H}_{n,j}}$. This gives:

$$\forall j \in \{1, ..., K\}, \forall x \in Par(\Omega_{\mathcal{H}_j}), \forall y \in \bigcup_{k=1}^{K} \Omega_{\mathcal{H}_k} : (y \succeq x \Rightarrow y = x). \tag{9}$$

Applying this time relation (2) in the backward direction demonstrates the announced result:

$$\forall j \in \{1, ..., K\} : Par(\Omega_{\mathcal{H}_{n,j}}) \subseteq Par(\bigcup_{k=1}^{K} \Omega_{\mathcal{H}_{n,k}}). \tag{10}$$

12

*Step 3*

By construction a terminal graph $\mathcal{H}_t$ contains only boundary vertices and TOVs. Now consider $x, y \in \Omega_{\mathcal{H}_t}, y \succeq x$, and consider a vertex $V_I$ in $\mathcal{H}_t$. If $V_I$ is a boundary vertex, then the variables in $I$ are already at their optimum bounds, and $x_I = y_I$. Otherwise, $V_I$ is a TOV and as in Step 2 above we can choose an $i_+ \in I \cap \mathcal{N}_+$ and $i_- \in I \cap \mathcal{N}_-$ for which the statement $y \succeq x$ implies both $y_{i_+} \geq x_{i_+}$ and $x_{i_-} \geq y_{i_-}$ and hence $x_I = y_I$. As the aggregates form a partition of the initial index set $\mathcal{N}$, the above immediately implies $x = y$, and hence, by relation (2), $x \in Par(\Omega_{\mathcal{H}_t})$. As this is true for any $x \in \Omega_{\mathcal{H}_t}$, we have demonstrated our result: $\Omega_{\mathcal{H}_t} = Par(\Omega_{\mathcal{H}_t})$.

### Example

In order to illustrate the working of the algorithm presented here we have implemented it in Mathematica$^{\mathrm{TM}}$ 9, using the built-in graph primitives [10]. The program generates random DAGs with a specified number of internal nodes, randomly chosen to be ascending or descending, using the layer method and then applies the algorithm. In Figure 2 we show an example of a typical input graph, and the output of the algorithm, in this case the union of three terminal graphs. Due to spatial limitations we cannot display the intermediate steps the algorithm makes to produce the final result.

## COMPLEXITY AND IMPROVED ALGORITHMS

We discuss here the complexity of different approaches and propose two further improvements of our algorithm. Complexity will be discussed relative to the number of variables $N$, and to the complexity of the Pareto front, as quantified by its number of faces $N_P$. A *face* is defined as a maximal convex subset of the Pareto front, which itself is a subset of the convex polytope $\Omega_{\mathcal{H}}$. Note that the number of faces of the Pareto front can be exponential in the number of variables, and that faces do not necessarily all have the same dimension.

The first improvement consist of introducing an additional contraction rule with forbidden steps, leading to a complexity linear in the number of dimensions of the inital problem, relatively to the size to Pareto front. This lead to resolution tree size $O(N.N_P)$, where the depth and the number of leaves of the tree respectively equal $N$ and $N_P$.
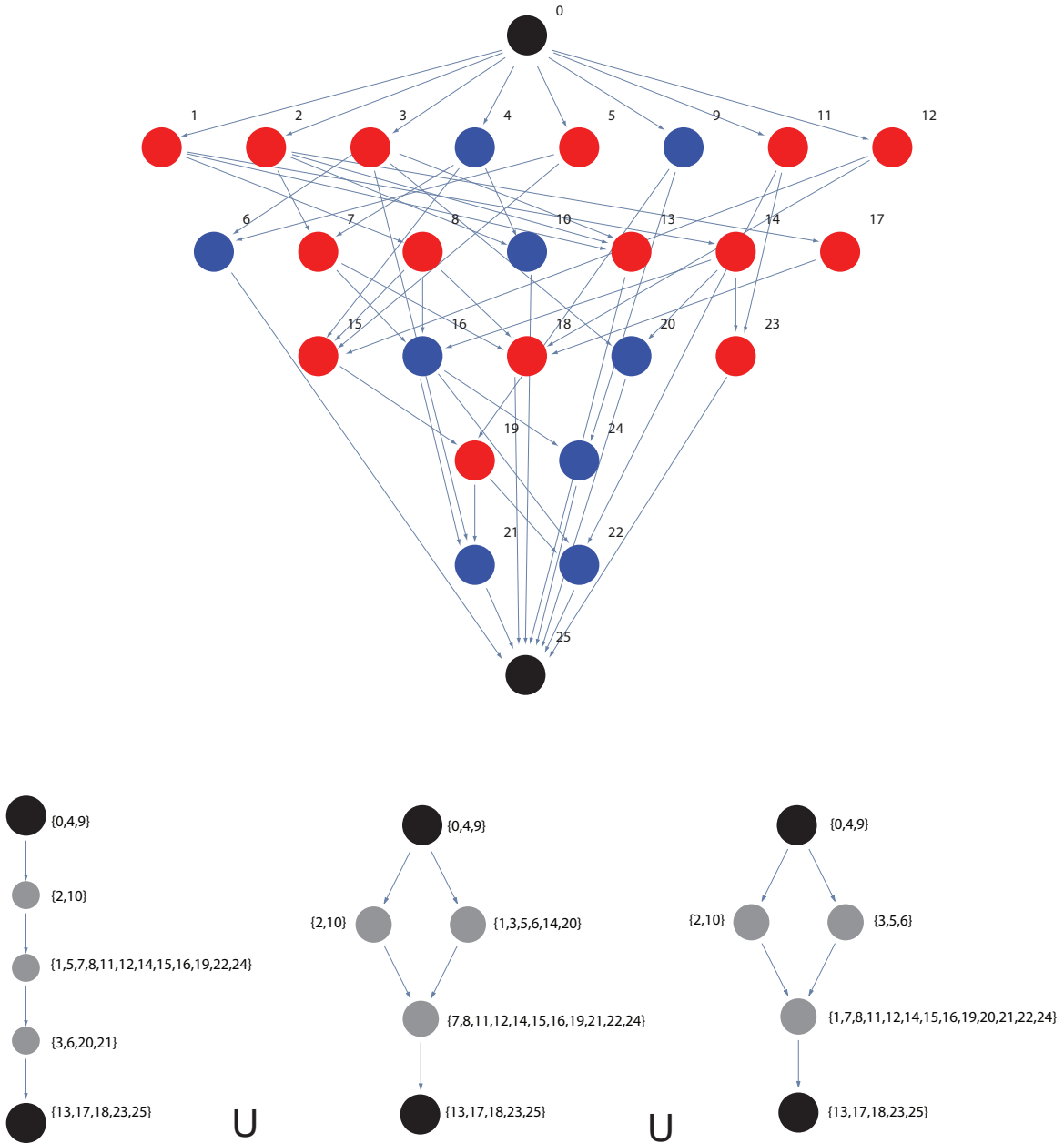
Figure 2: Top: A random DAG with $N = 24$ internal vertices, randomly chosen to be either ascending or descending. Bottom: The resulting Pareto optimal set obtained from our algorithm, consisting of the union of the set parameterized by three distinct terminal graphs. Colour code as in Figure 1, in addition, black circles represent boundary values.

The second improvement consist of a combinatorial description of the Pareto front from the set of solutions of specific components of the diagram. For this we will introduce the interface $\mathcal{I}$, which comprises all the conflicting vertices of the initial diagram. For each resolution of $\mathcal{I}$, the solutions of the monotone connected components can be computed independently, then assembled combinatorially. Parameterization by the different resolutions $\mathcal{I}$ can exponentially reduce the computing time and the size of the description of the full solution. In particular for series-parallel partial orders, the resolution of $\mathcal{I}$ is unique. Under the additional assumption that the size of monotone (or monochromatic) connected components is bounded, one obtains a resolution and a description of the Pareto front in $O(N)$, eventhough the Pareto front may comprise an exponential number of faces.

**Comparison of the complexity of different algorithms**

An exhaustive seach of the Pareto front would be to consider the $2^N$ corners of the $\{0,1\}^N$ hypercube, check for each whether the coordinates of the coresponding N-dimensional vector respect the partial order conditions (test up to $N(N-1)/2$ conditions), assess the Pareto optimality of each admissible corner relative the other admissible corners ($O(N_a^2)$, where $N_a$ is the potentially exponential number of admissible corners), and finally examine the Pareto optimality of all possible interpolations($O(2^{N_p})$) between the $N_p$ Pareto optimal admissible corners. Given that $N_a$ and $N_p$ can both scale like $2^N$, the complexity of this process can reach $O(2^{2^N})$.

The edge contraction algorithm as it is described so far provides several benefits compared to the exhaustive search. First, contraction operations on the Hasse diagram ensure to only explore solutions which are consistent with the partial order constraints.The complexity of the program can then be taken as examining the validity of subsets of contracted edges ($O(2^{N^2})$, as the number of edges is $O(N^2)$), then checking the resulting admissible subspacesfor their relative Pareto optimality ($O(2^{2N^2})$). Second, the restriction of contractions to vertices which are maximum within monotone connected components ensures that the final result of each contraction process is indeed Pareto optimal. Finally, each terminal diagram provides a full parametrization of a face of the Pareto front, the union of all which covers the full Pareto front, without requiring to assess Pareto optimality a second time within the set of potential solutions as Propositions 6and 5 would suggest in the general

case. The worst case complexity of the resolution tree is bounded by the a case where all possible subsets of edges have to be contracted and the number of edges is maximum, the first chosen vertex is in conflict with the $N-1$ other vertices, the second with $N-2$ vertices remaining after contraction, and so on, thus resulting in a complexity of $O(N!)$.

However, the version of the algorithm described so far has potential sources of increased complexity: (i) duplicated diagrams representing the exact same parameterization of a face of the Pareto front; (ii) diagrams $\mathcal{H}_i$ which aggregates the initial variables into a sub-partition of another diagram $\mathcal{H}_j$, so that the space of the Pareto front parameterized by $\mathcal{H}_i$ is only a subspace of the space parameterized by $\mathcal{H}_j$. Case (i) happens in particular when an extremal vertex conflicts with two or more other vertices, because the two corresponding contractions occur in a certain order along a resolution branch, and in another order along another branch. Case (ii) happens in particular when a vertex $V_0$ aims to a TOV $V_1$ and a conflicting vertex $V_2$: along a first resolution branch, $V_0$ aggregates with $V_1$, then $V_2$ aggregates with the resulting TOV, whereas along a second branch, $V_0$ aggregates with $V_2$, resulting in two distinct TOVs. Consequently, the parameterization given by two distinct TOVs in the second branch includes the solution obtained in the first.

A way to fix the redundancies due to case (i) would be to store known nodes of the resolution tree in a hash table, using a so-called *memoization* strategy [9]. We propose instead an improved version of the contraction rules, which ensures that every terminal diagram represents a distinct face of the Pareto front and which also removes sub-representations of the Pareto front due to case (ii). The resulting algorithm has a complexity of $O(N_P.N)$ where $N_P$ is the complexity of the Pareto front and $N$ the number of variables.

**Improved algorithm**

We define *frozen edges* as edges which cannot be contracted. Furthermore, we impose this property to be inherited downstream of the resolution tree, i.e. a frozen edge remains frozen after contraction of other edges. Otherwise, an edge is qualified as *free*. The improved version of the algorithm consists of modifying step 3 of the original algorithm as follows:

***Modified contraction rule***: If possible, contract extremal vertices $V_i$ which aim for

other vertices via a single free edge. Otherwise: (i) contract in priority $V_i$ with conflicting vertices, then with TOVs, and, (ii) for each $k = 2, ..., K$, the edges contracted to obtain $\mathcal{H}_{n,i}$, $i = 1, ..., k-1$, are frozen in $\mathcal{H}_{n,k}$.

As shown below, these modifications ensure that the terminal diagrams $\mathcal{H}_{N,t}, t = 1, ..., N_P$ describe all *distinct* faces of the Pareto front. Note that the total size of the resolution tree can be further minimized by treating in priority junctions comprising as few as possible alternatives, which generalizes the priority treament of single free edge contractions. However this changes neither the depth of the tree $N$, nor its number of leaves $N_P$.

**Proof of the improved algorithm**

An elementary point is that the algorithm can be consistently run until the leaf of each branch is reached. A potential issue would indeed be that the creation of a frozen edge leads to the existence of an extremal vertex not connected to any free edge. However, this never happens due to the priority contraction of single free edges, which implies that frozen edges are generated only at stages when every extremal vertex is connected to two or more free edges.

As the treatment of single free edge contractions does not differ from the rules of the initial algorithm, we set ourselves at a branching of the resolution tree corresponding to $k \geq 2$ alternative contractions of an extremal vertex $V_0$ aiming to $V_1, ..., V_k$, where the $j$ first $V_i$ conflict with $V_0$, and the remaining $V_i$ are TOVs, where $0 \leq j \leq k$.

We show first that the edge freezing rules leads to all admissible (in the sense of other rules) and distinct partitions of the initial variables. Consider the first contraction of $E(0 \to 1)$. The resulting graph $\mathcal{H}_{n,1}$ induces all admissible partitions such that $x_0$ and $x_1$ are in the same set. Consider then the contraction of $E(0 \to 2)$, where $E(0 \to 1)$ is frozen according to the modified algorithm. The resulting branch $\mathcal{H}_{n,2}$ induces all partitions such that $x_0$ and $x_2$ are in the same set but $x_1$ is not. For each $i$-th iteration of this process, the sub-tree stemming from $\mathcal{H}_{n,i}$ induces partitions such that $x_0$ and $x_i$ are in the same aggregate but $x_1, ..., x_{i-1}$ are not. Therefore, throughout the different branches, the contractions with $V_0$ enumerate without redundance all the accessible subsets of $\{x_0, ..., x_k\}$ containing $x_0$.

17

We now want to show that the face parameterized by every terminal diagram is embedded in a stricly different direction of the euclidian space, thus parameterizes a distinct face of the Pareto front. For this, we have to show that none of the partitions is included in another. This is obtained thanks to the prioritization of conflicting vertices contractions: when $V_0$ aggregates with a TOV $V_l$, $l > j$, any edge $E(0 \rightarrow i)$, $i \leq j$, joining $V_0$ to a conflicting $V_i$ is frozen in $\mathcal{H}_{n,l}$. Consequently, at this stage, none of the conflicting $V_i$ can be aggregated to the TOV resulting from the contraction of $E(0 \rightarrow l)$. This implies that two conflicting vertices susceptible to form a TOV cannot both aggregate to another TOV, at any stage of the process. Therefore, a TOV can only contain a single pair of conflicting variables, whereas at least two such pairs would be necessary to form a sub-partition of the aggregate.

**Graph interface and parameterized computation of the Pareto front**

We define the interface $\mathcal{I}$ of the initial problem $\mathcal{H}_0$ as the set of all conflicting vertices. $\mathcal{I}$ contains all extremal vertices of $\mathcal{H}_0$ which do not directly aim for the maximum or minumum bound. While $\mathcal{I}$ can be composed of several connected components, a monotone connected component of $\mathcal{H}_0$ may intersect several connected components of $\mathcal{I}$. We call $\mathcal{H}_u$ the diagrams obtained by aggregating first all the extremal vertices of $\mathcal{H}_0$ in $\mathcal{I}$. As all conflicting vertices have been aggregated into TOVs at this point, the algorithm only results in aggregation of extremal vertices with existing TOVs. In this sense, the remaining monotone connected components of each $\mathcal{H}_u$ are isolated from each other by TOVs.

Now call $\mathcal{C}_{u,v}, v = 1, .., V_u$ the montone connected components of $\mathcal{H}_u$ taken together with the TOVs they aim for. Each $\mathcal{C}_{u,v}$ can be solved separately, leading to its own set of leaves $\mathcal{C}_{u,v}^w, w = 1, ..., W_{u,v}$. The parametrizations of the different parts of the Pareto front of $\mathcal{H}_u$ can be obtained by concatenating all possible combinations of the $w$ indexes of the $\mathcal{C}_{u,v}^w$. Here the concatenation $\oplus$ between diagrams is defined as the merging of vertices which represent aggregated variable sets with a non-empty intersection. With these notations, we have:

$$Par(\mathcal{H}) = \bigcup_{u=1}^{U} Par(\mathcal{H}_u) \tag{11}$$

with for every $u$:

$$Par(\mathcal{H}_u) = \bigcup_{(w_1,...,w_v) \in W^\star} \left( \mathcal{C}_{u,1}^{w_1} \oplus ... \oplus \mathcal{C}_{u,v}^{w_v} \right) \tag{12}$$

18

where $W^\star = \{1, ..., W_{u,1}\} \times ... \times \{1, ..., W_{u,v}\}$.

Such a combinatorial representation of the Pareto front can be exponentially smaller than the number of faces of the Pareto front itself. In particular when the size of the $\mathcal{C}_{u,v}$ is bounded, the number of terms of the concatenation representing $Par(\mathcal{H}_u)$ increases linearly, while they represent an exponentially increasing number of faces of the Pareto front.

In the case of series-parallel $\mathcal{H}$, we have $U = 1$. This is due to series-parallel graphs being characterized by the absence of fence subgraph ("N" shaped motif). This property implies that there cannot be conflicting vertices which each participate to a junction. In other words, at least one of the two has no other alternative than contracting with its conflicting vertex, leading to the absence of branching process during the resolution of the interface of the Hasse diagram. Under the additional condition that monochromatic connected components have bounded size, all $\mathcal{C}_{1,v}$ have a bounded size, and the complexity of the resolution of these sub-diagrams is bounded. Thus the Pareto front of the full problem admits a representation which computation time grows linearly with the number of $\mathcal{C}_{1,v}$, which itself increases at much linearly with the number of initial variables.

## DISCUSSION AND CONCLUSION

We have described and demonstrated an algorithm which allows to find an exact parameterization of the Pareto front of any polytope defined by partial order relations within the hypercube $[0, 1]^N$, given an ideal point located at a corner of this hypercube. The solution is obtained in a linear number of steps with the number of parameters $N$, relative to the number $N_P$ of faces of the Pareto front.

This result is obtained by establishing a mapping between Hasse diagrams and polytopes, where a colouring of the graph encodes the location of the ideal point. More explicitly, vertices represent sets of aggregated variables, edges correspond to ordering relations, and colours correspond to the optimization objectives associated with the variables. Following a dynamic programming approach in the space of coloured graphs, the initial polytope is successively projected onto smaller dimension spaces, corresponding to edge contractions in its diagram representation. The Pareto front ultimately consists of the union of spaces corresponding to each terminal Hasse diagram obtained after $N$ contractions. A major advantage of this approach is that assembling the solution from the decomposed sub-problems

is a direct operation, which does not require any further Pareto computation than taking a union of sets.

We have furthermore introduced a parmeterized complexity approach, by introducing a specific subgraph, which we call the interface and which corresponds to the smallest set containing all the potential trade-offs. The edge contraction algorithm can be applied to this subgraph before all other vertices, until all conflicting parameters have been aggregated. This partial computation is sufficient to determine the dimension of the Pareto front. In particular, given that it is necessary to combine at least two nodes with conflicting objectives to generate a trade-off, the dimension of the Pareto front will be lower than $N_I/2$, where $N_I \leq N$ is the number of vertices in the interface. As soon as the interface is resolved, the diagram has a specific structure, where some coloured connected components are isolated from others. Due to this property, given a resolution of the interface, the Pareto front can be represented combinatorially from the solutions of the connected components which share a single objective. For series-parallel partial orders, the interface has a unique resolution. When, additionally, coloured connected components are of bounded size, the Pareto front, though of exponential complexity, can be computed and represented in $O(N)$.

The propositions of the section 2 of the article are generic and can be applied to reduce multi-objective optimization problems and re-assemble the Pareto front, similarly to other exact approaches to the Pareto set [6]. Specifically, the maximality property expressed in Proposition 3 and Proposition 4 shows how to decrease the search space while conserving the Pareto set, and Proposition 6 can then be used to break up the problem recursively into smaller subproblems.

In the context of the optimization of multi-input monotone systems, it would be important to investigate generalization of the algorithm developed here to linear inequality constraints or to ideal points other than corners of the search space.

* philippe.nghe@espci.fr

[1] Alber J, Fernau H, Niedermeier R (2004) Parameterized complexity: exponential speed-up for planar graph problems. *Journal of Algorithms* 52(1):26-56.

[2] Angeli D, Sontag ED (2003) Monotone control systems *Automatic Control, IEEE Transactions on* 48(10):1684-1698.

[3] Baykasoglu A, Owen S, Gindy N. (1999) A taboo search based approach to find the Pareto optimal set in multiple objective optimisation *Journ. of Engin. Optimization* 31:731-748.

[4] Carter M (2001) Foundations of mathematical economics, *The MIT Press, Cambridge MA*.

[5] Deb K (2001) Multi-Objective Optimization Using Evolutionary Algorithms *Wiley, New York.*

[6] Dhaenens C, Julien L, El-Ghazali T (2010) K-PPM: A new exact method to solve multi-objective combinatorial optimization problems *European Journal of Operational Research* 200:45-53.

[7] Ehrgott M, Gandibleux X (2000) An annotated bibliography of multi-objective combinatorial optimization *OR-Spektrum* 22(4):425-460

[8] Geilen M, Basten T, Theelen B, Otten R (2005) An algebra of Pareto points *Fundamenta Informaticae* 78(1):35-74

[9] Kambhampati S (2000), Planning Graph as a (dynamic) CSP: Exploiting EBL, DDB and other CSP search techniques in Graphplan *J. Artif. Intell. Res.* 12:1-34

[10] Belá Mulder, 2016, code available on request: `mulder@amolf.nl`.

[11] Policella N, Cesta A, Oddi A, Smith SF (2007). From precedence constraint posting to partial order schedules *Ai Communications* 20(3):163-180.

[12] Poelwijk FJ, de Vos MGJ, Tans SJ (2011) Tradeoffs and Optimality in the Evolution of Gene Regulation *Cell* 146:462-470.

[13] Pareto V (1971) Manuale di Economia Politica *Piccola Biblioteca Scientifica, Milan, 1906. Translated into English by Ann S. Schwier, Manual of Political Economy, MacMillan, London.*

[14] Schuetz R, Zamboni N, Zampieri M, Heinemann M, Sauer U (2012) Multidimensional Optimality of Microbial Metabolism *Science* 336:601-604.

[15] Shoval O, Sheftel H, Shinar G, Hart Y, Ramote O, Mayo A, Dekel E, Kavanagh K, Alon U (2012) Phenotype Space Evolutionary Trade-Offs, Pareto Optimality, and the Geometry of

Phenotype Space. *Science* 336:1157-1160.

[16] Skiena S (1990) Implementing Discrete Mathematics: Combinatorics and Graph Theory with Mathematica *Addison-Wesley, Reading, MA*.

[17] Sontag ED (2005) Molecular systems biology and control *European journal of control* 11(4):396-435.

[18] Voorneveld M (2003) Characterization of Pareto dominance. *Operations Research Letters* 31(1):7-11 .

[19] Yang P, Catthoor F (2003) Pareto-optimization-based run-time task scheduling for embedded systems *In Proc. Int. Conf. On Hardware/Software Codesign and System Synthesis* 120-125.

[20] Zitzler E, Thiele L, Laumanns M, Fonseca C, Fonseca VGD (2003) Performance assessment of multiobjective optimizers: An analysis and review. *IEEE Trans. on Evolutionary Computation* 7(2):117-132.