Partial design space exploration strategies applied in preliminary design

Thomas Richard de Latour, Raphaël Chenouard, Laurent Granvilliers

Partial design space exploration strategies applied in preliminary design

Thomas Richard de Latour[1], Raphaël Chenouard[1] and Laurent Granvilliers[1]

[1]LS2N, UMR6004, Nantes Université, École Centrale Nantes, CNRS, 44000 Nantes, France.

Contributing authors: thomas.richard-de-latour@ls2n.fr; raphael.chenouard@ls2n.fr; laurent.granvilliers@ls2n.fr;

Abstract

During preliminary phases in product design, on the basis of strong physical hypotheses (e.g. isotherm, steady state), physical and functional requirements can be expressed as coarse-grained constraint-based models on a few degrees of freedom, possibly including several design criteria to optimize. Such models are usually handled by multi-objective optimization solvers in order to find design solutions giving the best trade-offs between design criteria. Another approach developed in this paper is to partially explore all the areas of the design space using an anytime interval branch-and-prune algorithm called IDFS such that the design criteria are converted into so-called ε-constraints. The expected result is a sample of solutions diversified in both the objective space and the design space. Several quality indicators are introduced in order to measure this diversity and compare IDFS with two state-of-the-art multi-objective optimization solvers NSGA-II and NSGA-III on three real-world case studies. The results show that IDFS is able to identify new close-to-optimal designs and permits a better understanding of the design space. This framework provides a promising alternative tool for decision making, in particular for integrating interaction in the preliminary design process.

Keywords: Design space exploration, numerical constraint satisfaction problem, anytime branch-and-prune algorithm, multi-objective optimization, preliminary design

1 Introduction

1.1 Context

New challenges in the preliminary design process of products or systems are to be able to deal with their increasing complexity as well as considering their whole life cycle [1]. In this context, defining the best system regarding design requirements is a hard task where multiple criteria can be used to analyze possible design candidates. Design space exploration [2] addresses this problem by automating the selection of design alternatives prior to any detailed design phases. At this stage of the design process, a system can be represented by a coarse-grained model assuming a minimum number of degrees of freedom and strong physical hypotheses (isotherm, steady state, etc.) while focusing on main peculiarities of the system. The unknowns can be defined by variables lying in domains and the design requirements can be expressed as constraints on these variables, i.e. we have a constraint satisfaction problem (CSP) [3–6] that must be small-scale with at most a few dozens of variables. Moreover, the functional, physical, or topological specifications of a system often lead to several design criteria to optimize, e.g. maximize

power and minimize cost or impact. Hence, designers seek to compute the best trade-off solutions while satisfying all constraints.

Optimization methods are widely used in this context to compute the set of best trade-off solutions in the objective space called *Pareto Front* (PF) [7–12]. On the one hand, the different criteria to optimize in a design problem can be transformed into a single objective function using different techniques, such as the weighted sum technique where all criteria are weighted and aggregated or the ε-constraint method where all criteria but one are relaxed as inequality constraints. It comes in general a mixed-integer non-linear optimization problem (MINLP) that can be handled by global optimization solvers in order to compute a guaranteed optimum located on the PF. This approach is particularly effective for evaluating extreme points of the PF that can be useful to get a first understanding of the solution set. Unfortunately, multiple runs are needed to approximate the PF, which quickly becomes laborious, especially in high dimensional objective spaces. On the other hand, it is possible to directly solve a mixed-integer multi-objective optimization problem (MOP) using meta-heuristics like evolutionary algorithms, simulated annealing, or particle swarm optimization. These techniques rely on non-deterministic algorithms to compute optimal or quasi-optimal solutions in a reasonable time [12, 13].

1.2 Motivating example and limits

To illustrate our context, we consider the design of an internal combustion engine (*ice*) described in [14]. This problem is modeled as a simple flat head combustion chamber (Fig. 1). It is based on thermodynamic principles with the hypothesis of an ideal thermal efficiency. The corresponding problem has five continuous design variables and nine inequality constraints (Annex A.1). The initial design objective is to maximize the obtained power per unit displacement volume BKW/V (kW/l). We consider here a second criterion in order to minimize the fuel consumption $ISFC$ (g/kWh).

The *ice* model is a bi-objective non-linear continuous optimization problem which can be solved by the genetic algorithm NSGA-II [15] using the

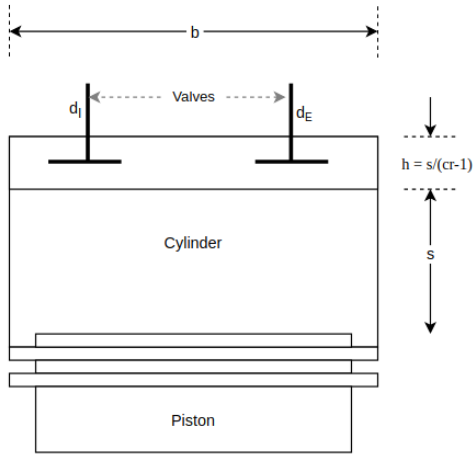


Fig. 1: Internal combustion engine design problem from [14]

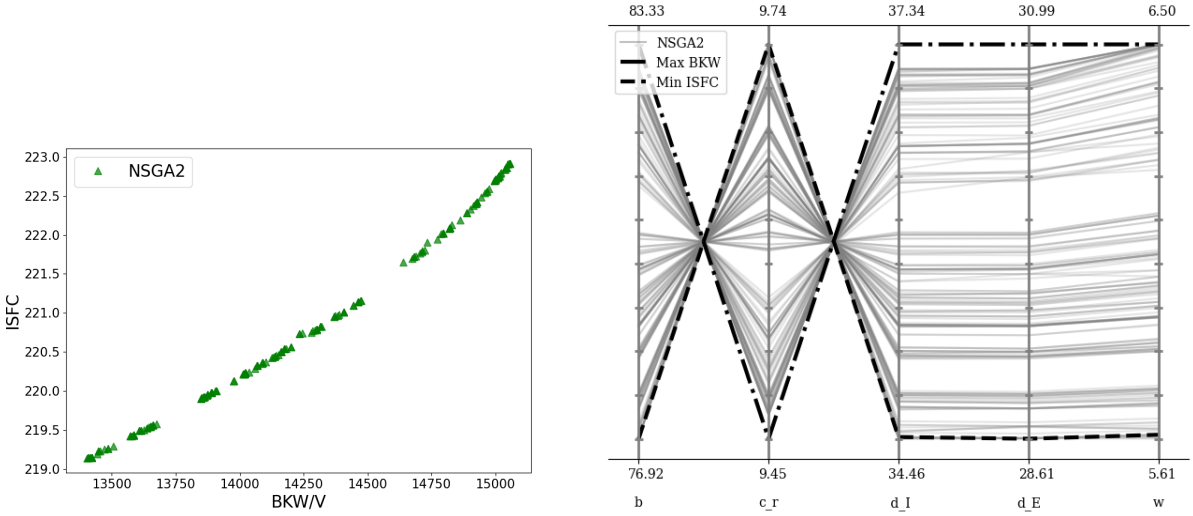
Table 1: *ice* model main parameters

Name	Description	Variable type
b	Cylinder bore, (mm)	Design
C_r	Compression ratio	Design
d_E	Exhaust valve diameter (mm)	Design
d_I	Intake valve diameter (mm)	Design
w	Revolutions per minute at peak power	Design
ISFC	Fuel consumption (g/kW-h)	Objective
BKW/V	Brake power per volume unit (kW/1)	Objective

pymoo library [1]. The solving parameters used are a population size of 200, a tolerance on the objectives of 10^{-5}, and at most 1500 generations. A good convergence is obtained after 250000 function evaluations. The approximated Pareto front Θ_{ice} composed of 190 solutions is presented in the *objective space* (Fig. 2a). The corresponding solutions are also represented in the *design space* (Fig. 2b). Those designs reflect the best alternatives of our problems in a mathematical sense. However, knowing the limits of this model, the designer may want to consider the full picture of the feasible space and seek different alternatives in the design space.

In general, some limits can be observed in the use of optimization during the preliminary design phase. Many hypotheses still last : steady state, approximate data and uncertain values at the frontier of the system [16]. As a result, the computed solutions may not accurately reflect the expected performances while they are often

[1] https://pymoo.org/



(a) Approximated Pareto front of the *ice* problem obtained with NSGA-II.

(b) Solutions from NSGA-II presented in the design variable space.

Fig. 2: Results from the optimization of the *ice* problem with NSGA-II

located at the frontier of the feasible space with multiple active constraints. Secondly, objectives may be difficult to express regarding all aspects (functional, usage, manufacturing, and environmental) of complex system design. In particular, environmental specifications often imply several additional criteria [17]. Thirdly, optimal or near-optimal solutions may not reflect the shape of the design solution space and interesting sub-optimal candidate designs may be out of reach. Even if MOP solvers are able to compute many diverse solutions, their distribution is often sought only in the objective space [18–20]. Furthermore, the performances of those solvers are degraded in the presence of several objectives (3 or more) [21, 22].

1.3 Propositions

Considering those observations, this paper proposes a new methodology based on a partial exploration of the design space modeled as a CSP such that each design criterion is handled as an ε-constraint. The goal is to compute a reasonably sized subset of solutions that are diverse enough to cover the solution space in both the design space and the objective space. From a decision-making perspective, this approach enables to get a clear vision of the feasible space, to position the final design relatively to other alternatives and to favor an interactive method with the user. To this end, we propose to implement a deterministic branch-and-prune (B&P) algorithm based on interval methods [23] with an anytime behavior as done in [24]. Our contribution is threefold:

- We introduce a new search strategy for anytime B&P algorithms with good scalable properties to tackle design CSP.
- We define new quality indicators to evaluate the convergence towards a set of diverse solutions in both the design space and the objective space.
- We evaluate our partial exploration paradigm by comparing it with well-known evolutionary optimization methods on three case studies.

The rest of this paper is structured as follows. Section 2 introduces preliminary notions essential for the understanding of the mechanisms behind B&P algorithms and optimization solvers. Section 3 details previous works related to anytime B&P algorithms, as well as the related contributions. Experiments are conducted in Section 4. The experimental protocol is detailed in the first part of this section, then each case is treated separately. The results are discussed in section 4.5 followed by a conclusion in Section 5.

2 Preliminaries

In this paper, engineering design problems are modeled as numerical constraint satisfaction or optimization problems involving continuous variables, discrete variables, and various kinds of constraints like nonlinear equations, inequality constraints, and global constraints. We will introduce these problems and associated solving methods thereafter.

2.1 Intervals

Let $[x] = [\underline{x}, \overline{x}]$ denote a closed interval of real numbers and let $\mathbb{I}$ be the set of intervals. The *width* of $[x]$ is defined by $w([x]) = \overline{x} - \underline{x}$. The *hull* of a set $S \subseteq \mathbb{R}$ is defined by the interval $\Box S = [\inf S, \sup S]$.

A *box* $[\mathbf{x}]$ of dimension n is a Cartesian product of intervals $[x]_1 \cdots \times [x]_n$. The *hull* of a set $S \subseteq \mathbb{R}^n$ is defined by the box $\Box S = \Box S_1 \times \cdots \times \Box S_n$ where each S_i is the i-th projection of S. The *hull* of a set of boxes corresponds to the hull of their union.

2.2 Constraint satisfaction problems

Definition 1 (CSP) A (numerical) *constraint satisfaction problem* is a triple $\mathcal{P} = (\mathbf{x}, [\mathbf{x}]^0, C)$ where

- $\mathbf{x} = (x_1, \ldots, x_n) \in \mathbb{R}^n$ is a vector of variables such that we have $x_i \in \mathbb{Z}$ for each i in a given a set of indices $\mathcal{I} \subseteq \{1, \ldots, n\}$,
- $[\mathbf{x}]^0$ is a box of dimension n such that $[x]_i^0$ is the domain of x_i for each $i \in \{1, \ldots, n\}$, and
- C is a set of constraints over $\mathbf{x}$ such that each constraint restricts the acceptable values of the variables taken in $[\mathbf{x}]^0$.

A *solution* of $\mathcal{P}$ is a tuple $\mathbf{s} = (s_1, \ldots, s_n) \in [\mathbf{x}]^0$ such that we have $s_i \in \mathbb{Z}$ for each $i \in \mathcal{I}$ and each constraint $c \in C$ is satisfied by $\mathbf{s}$, i.e. c is true when each x_i is assigned to s_i. The set of all the solutions of $\mathcal{P}$ is denoted by $\mathcal{S}$.

Numerical CSPs can be solved in a *complete* way by spatial branch-and-prune algorithms that calculate a set of boxes enclosing $\mathcal{S}$ at a given precision [23]. To this end, a relative precision is introduced for each variable, which may reflects e.g. physical requirements in an engineering design context. Given $\boldsymbol{\epsilon} = (\epsilon_1, \ldots, \epsilon_n) \in \mathbb{R}_+^n$, we say that a box $[\mathbf{x}] \in \mathbb{I}^n$ is an $\boldsymbol{\epsilon}$-*box* if we have

$$w([x]_i) \leq \epsilon_i(|\underline{x}_i| + |\overline{x}_i|)$$

for each $i \in \{1, \ldots, n\}$, i.e. the box is precise enough with respect to the requirements. An $\boldsymbol{\epsilon}$-*paving* is a set of $\boldsymbol{\epsilon}$-boxes enclosing $\mathcal{S}$.

A branch-and-prune algorithm generates a search tree from the initial box $[\mathbf{x}]^0$ until reaching an $\boldsymbol{\epsilon}$-paving. Every box $[\mathbf{x}]^k$ of the tree is contracted (or pruned, or reduced) such that the new box $[\mathbf{x}]^{k+1}$ verifies

$$[\mathbf{x}]^k \cap \mathcal{S} \subseteq [\mathbf{x}]^{k+1} \subseteq [\mathbf{x}]^k,$$

i.e. no solution in $[\mathbf{x}]^k$ is lost. After this step, an empty box is fathomed, an $\boldsymbol{\epsilon}$-box is labelled as a solution box, and any other box is split into a set of sub-boxes. In order to ensure the convergence of the algorithm, it suffices to regularly split the largest component of every box.

This algorithm requires an exponential time cost in the worst case. So, its application must be restricted to small-scale problems or medium-scale problems with specific structures. Moreover, the use of powerful contraction methods is of particular importance in order to accelerate the convergence. Given a box $[\mathbf{x}]^k$, constraint propagation is a fixed-point algorithm that reduces $[\mathbf{x}]^k$ by considering the constraints one by one. Each constraint is processed by a reduction technique like HC4 [25] and the modifications of domains are propagated to the other constraints, and so on. It is expected to converge towards the hull of the set $[\mathbf{x}]^k \cap \mathcal{S}$ with reductions that are local in essence, which may be optimistic in many situations due to the form of nonlinear constraints. A complementary approach is to generate a linear relaxation of the constraints using e.g. Taylor forms [26]. Then each of the $2n$ facets of $[\mathbf{x}]^k$ can be reduced by solving a linear program. The quality of reductions strongly depends on the tightness of relaxations. Such techniques are implemented in the IBEX solver [27] that is also our experimentation platform.

Example 1 Let $x_1^2 + x_2^2 \leq 2$ be a constraint and let $[\mathbf{x}] = [-5, 5] \times [-5, 5]$ be a box. Using interval arithmetic, we deduce that $x_2^2 \in [0, 25]$ and it follows that $x_1^2 \leq 2$. The domain of x_1 is then reduced

to $[-\sqrt{2}, \sqrt{2}]$. The HC4 technique automatizes such interval reasoning and the reduced box here is the hull of the circle of radius $\sqrt{2}$ centered at $(0, 0)$.

2.3 Optimization problems

Optimization problems are basically CSPs with one or several objective functions. In the following, let $\mathcal{S}$ be the solution set of a CSP as previously introduced, also called *feasible set* in an optimization context.

Definition 2 (MINLP) Given an objective function $f : \mathbb{R}^n \to \mathbb{R}$, a *mixed-integer nonlinear program* can be defined as

$$\min_{\mathbf{x} \in \mathcal{S}} f(\mathbf{x}).$$

A solution $\mathbf{x}^* \in \mathcal{S}$ is *optimal* if for all $\mathbf{x} \in \mathcal{S}$ we have $f(\mathbf{x}^*) \leq f(\mathbf{x})$. The value $f(\mathbf{x}^*)$ at any optimal solution $\mathbf{x}^*$ is the *global optimum*.

Spatial branch-and-bound algorithms are able to globally solve MINLPs in a *deterministic* way. They are basically designed as branch-and-prune algorithms with bounding components. Lower bounds of global optima can be obtained through the solving of linear relaxations. Feasible points and upper bounds of global optima are derived in general by local solvers implementing e.g. classical mathematical methods. Once again, this algorithm runs in an exponential time worst case and it may fail to solve large-scale problems. BARON [28] and ANTIGONE [29] are powerful solvers in this domain.

Definition 3 (MOMINLP or simply MOP) Given an integer $p \geq 2$ and p functions $f_1, \ldots, f_p : \mathbb{R}^n \to \mathbb{R}$, a *multi-objective mixed-integer nonlinear program* can be formulated as

$$\min_{\mathbf{x} \in \mathcal{S}} (f_1(\mathbf{x}), \ldots, f_p(\mathbf{x})).$$

Given two solutions $\mathbf{x}, \mathbf{y} \in \mathcal{S}$, we say that $\mathbf{x}$ *dominates* $\mathbf{y}$ if we have $f_k(\mathbf{x}) \leq f_k(\mathbf{y})$ for each $k \in \{1, \ldots, p\}$ and $f_k(\mathbf{x}) < f_k(\mathbf{y})$ for some $k \in \{1, \ldots, p\}$. Any solution $\mathbf{x}^* \in \mathcal{S}$ is *Pareto optimal* if no other solution dominates it. The *Pareto set* is the set of all Pareto optimal solutions. The *Pareto front* is the set of all vectors of the form $(f_1(\mathbf{x}^*), \ldots, f_p(\mathbf{x}^*))$ such that $\mathbf{x}^*$ is Pareto optimal.

A MOP maps the *decision space* of dimension n to the *objective space* of dimension p. Solving MOPs is hard when the number of objectives p is increased and they are usually handled by metaheuristics. These methods aim at finding a good approximation of the Pareto set that is distributed as well as possible in the Pareto front. For example, population-based methods like ant-colony optimization and evolutionary algorithms maintain a set of candidates in the decision space. A new generation of candidates results from crossover and mutation, and the challenge is to ensure *convergence* towards good (and even Pareto optimal) solutions and to maintain *diversity* of the solutions. In other words, one wants the candidates to move closer to the Pareto front and to be located in different areas of the Pareto set and the Pareto front.

Scalarization techniques reformulate MOPs as MINLPs. For example, the weighted-sum technique assigns weights to the objective functions and aggregates them into one single function, and the ε-constraint method transforms all objectives but one into inequality constraints [30]. Solving those MINLPs globally aims at finding non dominated solutions.

Many performance indicators have been introduced over the years to evaluate the quality of Pareto set approximations. [31] classifies them in categories and motivates their use to compare different algorithms and to define stopping criteria. For example, the hypervolume, related to convergence and distribution properties, is the volume of the dominated space between the approximated Pareto front and a reference point [32]. The higher the hypervolume is, the better the approximation.

2.4 Problems in engineering design

In this paper, we consider the design process based on three main steps as depicted in Fig. 3. First, the needs and requirements analysis aims at identifying the decision criteria to consider, as well as the design variables and performance or operational constraints to satisfy. Secondly, the design problem is formulated as a mathematical decision problem used to explore the design space to support designers in the selection of the best promising principles of design solutions. Finally, the selected principles of solutions are developed for a detailed analysis.

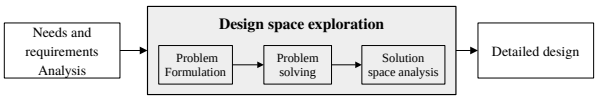


Fig. 3: Simplified design process.

We focus on the design space exploration process, also called design synthesis process. This step is often adapted to use a specific optimization solving technique [33, 34]. In this case, the design decision problem is reformulated into a design optimization problem. As previously introduced, the criteria of the decision problem may be transformed into ε-constraints, or aggregated into one or several objective functions, depending on the used solving technique. In this case, the major reason guiding the reformulation is to fit with the solving technique requirements, like efficiently handling constraints (equations, inequalities, table constraints, etc.) and supporting a single or several objective functions.

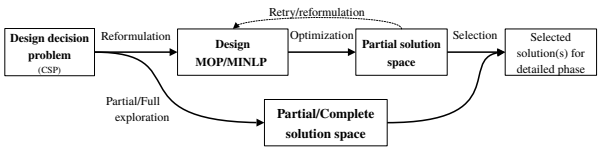


Fig. 4: Design exploration steps.

Fig. 4 shows the two main possible paths followed by designers to explore the design space. In the first one, the result is a partial solution space that can be limited to a single solution for a single optimization problem. Then, several reformulations can be applied to get more diverse solutions. The optimization problem can be modified to aggregate the criteria differently or to adapt the ε-constraints until the computed solution fits the expectations. In the second path, designers directly use a solving technique able to explore the design decision problem, as for instance a CSP solver [35]. The main issue, in this case, is that the solution space is often huge and not reasonable to fully explore. Most exploration strategies do not consider any diversity indicator to guide the exploration. Then, a partial solution set generated by a default strategy may correspond to similar principles of design solutions. A few exploration strategies were introduced to compute partial solutions based on an anytime branch-and-prune algorithm and a distance indicator [24], but the computational complexity does not allow scaling to the design of complex systems.

3 Anytime deterministic search

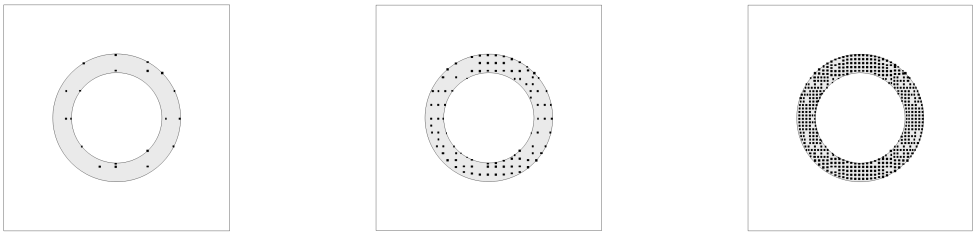
3.1 Quality of approximation

The solution set $\mathcal{S}$ of a (numerical) CSP is in general a union of disconnected regions in $\mathbb{R}^n$ and calculating an $\boldsymbol{\epsilon}$-paving of $\mathcal{S}$ is not practicable in general. From an application perspective, it could be desirable to sample those regions, hence providing solutions with different characteristics and revealing the shape of $\mathcal{S}$. We propose to handle this problem using a partial deterministic search algorithm, more precisely a branch-and-prune algorithm with limited resources like a time limit or a limit on the number of solutions. In this algorithm, the selection component of the next node of the search tree to be processed controls the exploration strategy. Let us examine the following academic example.

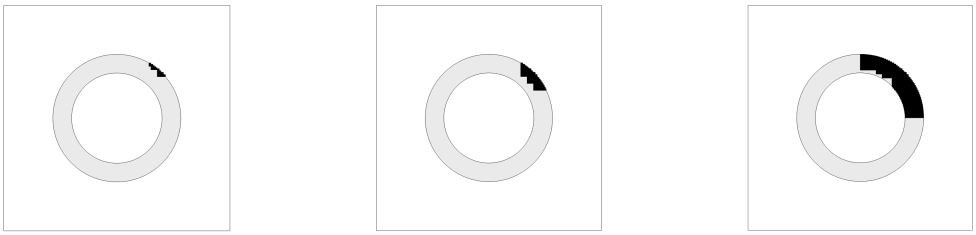
Example 2 Let $\mathcal{P} = (\mathbf{x}, [-5,5]^2, \{2 \leq x_1^2 + x_2^2, x_1^2 + x_2^2 \leq 4\})$ be a CSP. Its solution set $\mathcal{S}$ corresponds to the space between two concentric circles, as depicted in Fig. 5. A depth-first search strategy goes down each branch of the search tree as deep as possible before it backtracks to the closest unexplored node. As a consequence, the computed $\boldsymbol{\epsilon}$-boxes are accumulated in the same sub-region of $\mathcal{S}$. A second strategy is able to generate a more diverse set of solutions at any time of the computation.

As illustrated above, our goal is to design an anytime algorithm that is able to return a good approximation of $\mathcal{S}$ when the allocated resources are exceeded. The quality of such an approximation can be observed according to the following criteria inspired from the performance indicators used in the multi-objective optimization field.

- *Spread* Q_H. The computed approximation intersects the different regions of the solution set and some $\boldsymbol{\epsilon}$-boxes are close to their boundaries.
- *Distribution* Q_A. The $\boldsymbol{\epsilon}$-boxes are well distributed in the solution set.



(a) 20 ϵ-boxes (b) 100 ϵ-boxes (c) 500 ϵ-boxes



(d) 20 ϵ-boxes (e) 100 ϵ-boxes (f) 500 ϵ-boxes

Fig. 5: Partial covering of the solution set of $\mathcal{P} = (\mathbf{x}, [-5,5]^2, \{2 \leq x_1^2 + x_2^2, x_1^2 + x_2^2 \leq 4\})$. Figures from a) to c) are obtained with an anytime search strategy and, from d) to f) with a depth-first search strategy.

- *Cardinality* Q_C. The number of ϵ-boxes shows the capacity of the algorithm to find solutions.

The first two criteria are defined below. Let $\Sigma = \{[\mathbf{s}]^1, \ldots, [\mathbf{s}]^m\}$ be a set of ϵ-boxes approximating $\mathcal{S}$ and let $\square\Sigma$ be the hull of Σ. Given the initial box $[\mathbf{x}]^0$, the *spread* indicator is a ratio between the volume of the approximation and the volume of the initial box, defined by

$$Q_H = \frac{v(\square\Sigma)}{v([\mathbf{x}]^0)} \tag{1}$$

where the volume of a box $[\mathbf{x}]$ is defined by

$$v([\mathbf{x}]) = \prod_{j, w([x]_j)>0} \frac{w([x]_j)}{\epsilon_j}.$$

The above definition considers only the intervals that are not reduced to one point. That eliminates each variable having the same value in all the ϵ-boxes. This notion is illustrated by Fig. 6.

The *distribution* indicator is a mean minimal distance between the ϵ-boxes and we expect that they are not too close to each other. This indicator

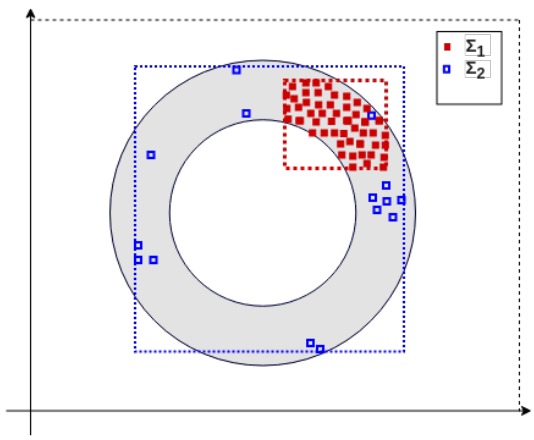


Fig. 6: Σ_1 and Σ_2 are two approximations of the solution space $\mathcal{P} = (\mathbf{x}, [-5,5]^2, \{2 \leq x_1^2 + x_2^2, x_1^2 + x_2^2 \leq 4\})$ and we have $Q_H\ (\Sigma_1) < Q_H\ (\Sigma_2)$.

is defined by

$$Q_A = \frac{1}{m} \cdot \sum_{i=1}^{m} \min\{d([\mathbf{s}]^i, [\mathbf{s}]^j) : 1 \leq j \leq m, j \neq i\}$$

where d is a distance function. The Hausdorff distance between two intervals is defined by

$$d_h([x], [y]) = \max(|\underline{x} - \underline{y}|, |\overline{x} - \overline{y}|)$$

and it is extended to boxes by taking the maximum distance componentwise. When used in Q_A it tends to favor diversity along the largest domains. To counteract this effect, we define the *normalized Hausdorff distance* between boxes

$$\overline{d_h}([\mathbf{x}], [\mathbf{y}]) = \max \left\{ \frac{d_h([x]_i, [y]_i)}{w([x]_i^1)} : 1 \leq i \leq n \right\}$$

where $[\mathbf{x}]^1$ is the box obtained by contraction of the initial box $[\mathbf{x}]^0$. By this way, the variables with the largest domains after the first contraction are penalized. Compared to other metrics, the Hausdorff distance does not add different quantities (like power and emissions), this is the interval equivalent of the Tchebytcheff norm used in [36].

3.2 Algorithm

Best-first search (BestFS) is a search strategy on graphs [37]. During the search, the next candidate nodes are evaluated using a merit or measure-of-best function ρ and the best one is selected to continue the exploration. For example, this strategy is classically implemented in branch-and-bound algorithms for globally solving optimization problems such that next node gives the smallest (or greatest depending on the optimization direction) lower bound of the objective function. Anytime branch-and-prune strategies can be built on this paradigm with merit functions designed to maximize the quality of the resulting set of $\boldsymbol{\epsilon}$-boxes. The difficulty is here to address both convergence and diversity properties. As observed in [24, 38], BestFS strategies lack efficiency to converge quickly towards the solutions since they tend to mimic breadth-first exploration. To overcome this problem, it is interesting to hybridize best-first and depth-first stages, BestFS favouring the diversity and DFS ensuring the convergence. Distant-most and depth-first-search (DMDFS) [24] is such an hybrid strategy such that the merit function ρ leads to maximize the distance between the $\boldsymbol{\epsilon}$-boxes already computed and the unexplored part of the search space. While resulting in good approximations, the evaluation of ρ is very expensive and grows with the number of $\boldsymbol{\epsilon}$-boxes.

Alg. 1 is an anytime branch-and-prune algorithm designed as an hybrid BestFS-DFS strategy. The next selected box (line 4) is the front element of the list of unexplored boxes L. The DFS behavior follows from the management of L as a stack such that the sub-boxes generated by the splitting step are inserted in the front of L (lines 13 and 14). The BestFS behavior consists in sorting L based on ρ each time a solution is found (line 10). We propose to use a cheap and efficient merit function such that L is sorted in descending ordering of the depth of nodes in the search tree. By this way, the node selected at the next iteration of the loop is the highest in the search tree, which is expected to be the biggest and least explored region of the search space. It follows a hybrid path between DFS and bread-first exploration, as inspired from [39]. The acronym used in [39] to designate this strategy is kept here, namely interleaved depth-first search (IDFS).

Algorithm 1 Anytime B&P

Input: CSP $\mathcal{P} = (\mathbf{x}, [\mathbf{x}]^0, C)$, precision vector $\boldsymbol{\epsilon}$, measure-of-best function ρ, stopping criterion ϕ
Output: set of $\boldsymbol{\epsilon}$-boxes Σ approximating the solution set of $\mathcal{P}$

1: **let** $L \leftarrow ([\mathbf{x}]^0)$ be a list of boxes
2: **let** $\Sigma \leftarrow \emptyset$ be a set of boxes
3: **while** L is not empty and ϕ is false **do**
4: **let** $[\mathbf{x}]^k$ be the front element of L
5: remove $[\mathbf{x}]^k$ from L
6: $[\mathbf{x}]^{k+1} \leftarrow$ contract $[\mathbf{x}]^k$ using C
7: **if** $[\mathbf{x}]^{k+1}$ is not empty **then**
8: **if** $[\mathbf{x}]^{k+1}$ is an $\boldsymbol{\epsilon}$-box **then**
9: insert $[\mathbf{x}]^{k+1}$ in Σ
10: sort L based on ρ
11: **else**
12: $([\mathbf{x}]^{k+2}, [\mathbf{x}]^{k+3}) \leftarrow$ split $[\mathbf{x}]^{k+1}$
13: push $[\mathbf{x}]^{k+2}$ in the front of L
14: push $[\mathbf{x}]^{k+3}$ in the front of L
15: **end if**
16: **end if**
17: **end while**

4 Case Studies

The partial design space exploration approach is applied to three real-world case studies, an

internal combustion engine (*ice*) [14], an electromechanical actuator (*act*) [7], and the water problem (*wat*) from [40], and it is compared with optimization solvers. Their CSP and MOP models are given in the appendices.

4.1 Protocol

The design space exploration is done using an anytime B&P algorithm implemented in C++ with the IBEX[2] library. Our IDFS strategy considers a largest-first branching operator that always selects the largest component of a box and a propagation algorithm based on HC4 as reduction technique. The relative precision is assigned to $\epsilon_i = 10^{-3}$ for each variable x_i. The stopping criterion of the partial exploration is defined as the number of $\boldsymbol{\epsilon}$-boxes solutions Q_C, which is sized to find a good compromise between diversity in the solution space and a reasonable number of designs to consider. To estimate Q_C, a first exploration is done with an oversized stopping criterion $Q_C^0 \geq 1000$ to analyze the evolution of Q_H and Q_A. Then Q_C is built using the formulas

$$\frac{Q_H(m)}{Q_H^{max}} \geq 80\%, \tag{2}$$

$$\frac{Q_A(m+\Delta m) - Q_A(m)}{Q_A(m) - Q_A(Q_C^0)} \leq 1\%, \tag{3}$$

with $1 \leq m \leq Q_C^0$, Δm a variation of the number of solutions and Q_H^{max} the volume of the hull containing the feasible space. The first formula states that the subset of $\boldsymbol{\epsilon}$-boxes has to cover at least 80% of the complete feasible space area. The second one estimates when a variation of Δm solutions does not decrease Q_A by more than 1%. This second step can be seen as the point where IDFS starts to increase the density of solutions in areas already explored enough. Eventually, we analyse the global evolution of Q_H and Q_A to conclude. The final subset of Q_C $\boldsymbol{\epsilon}$-boxes distributed in the feasible space is denoted by Σ.

The multi-objective optimization solvers used are the evolutionary algorithms NSGA-II and NSGA-III through the python library pymoo[3]. NSGA-II is preferred for the bi-objective problem, and NSGA-III for MOPs with 3 or more objectives. All the three case studies are under-constrained problems and feasible solutions are obtained early in the solving process. The challenge is to converge as close as possible to the Pareto front. To ensure a good convergence, the resolution is evaluated through the hypervolume H_V [32] with the Nadir point as reference point. Both algorithms lead to a subset of solutions approximating the Pareto front denoted by Θ. Table 2 presents the resolution parameters for NSGA-II and NSGA-III.

Table 2: NSGA-II and NSGA-III shared resolution parameters, with n the number of variables of the problem

Parameter	Value
Mutation rate	1/n
Mutation distribution	30
Crossover probability	0.95
Crossover distribution	20

Once a problem is solved with both exploration and optimization approaches, the results are compared according to Q_H, Q_A and H_V. Each step of the experiment is detailed for the first problem *ice*. All the experiments have been done with a Linux Core i7-9850H 2.6 GHz (16 GB).

4.2 Internal combustion engine

4.2.1 Experiments

The *ice* problem described in the introduction is first solved through an optimization method using NSGA-II. The normalized hypervolume is given in Fig. 7. The solving process takes 25 seconds in average on 20 different runs with random seeds.

This problem is simple enough in order to clearly illustrate the partial exploration paradigm, as follows. The design criteria are relaxed as follows:

$$BWK \geq 13400,$$
$$ISFC \leq 223.2.$$

A first exploration using IDFS with $Q_C^0 = 2000$ is done to study the evolution of the quality of

[2] http://www.ibex-lib.org/
[3] https://pymoo.org/

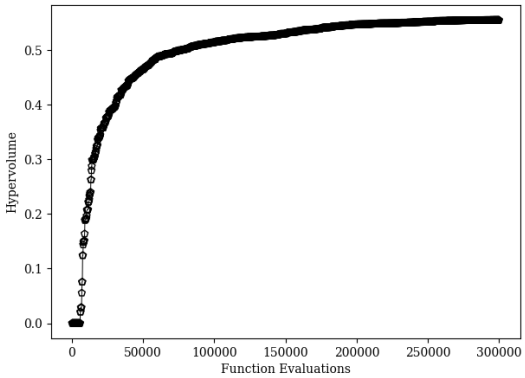
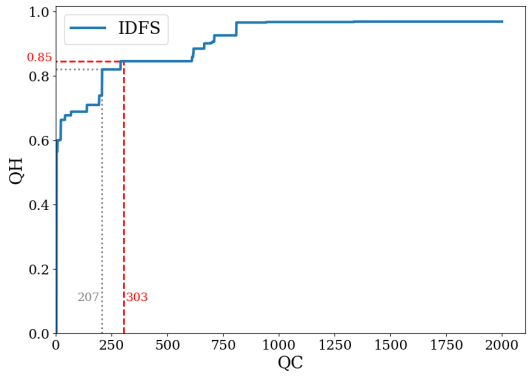
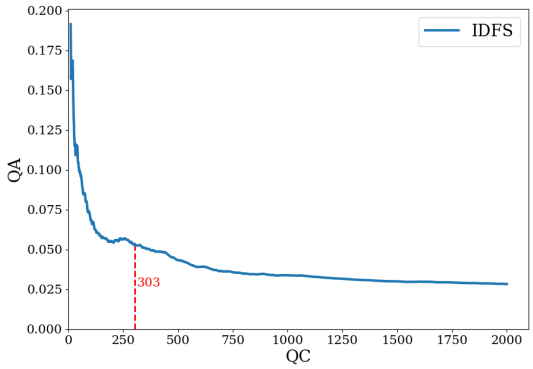


Fig. 7: Hypervolume indicator for the optimization of *ice* with NSGA-II.



(a) QH quality indicator computed with IDFS strategy



(b) QA quality indicator computed with IDFS strategy

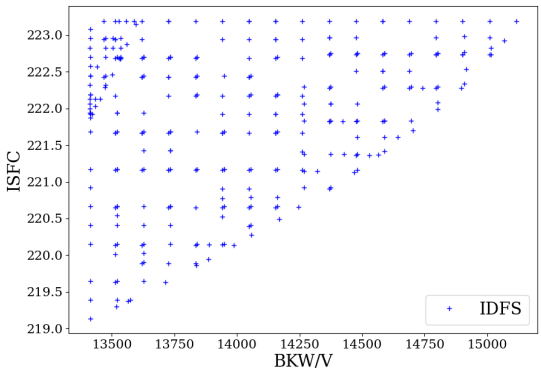
Fig. 8: Quality indicators evolution when solving *ice* with IDFS strategy.

the approximation and to estimate the right stopping criterion. Fig. 8a and Fig. 8b respectively present the overall spread Q_H and the mean minimal Hausdorff distance Q_A. The evolution of Q_H shows that after 207 computed solutions, at least 80% of the hull of the solution space is covered. Q_H can be improved to 85% by slightly increasing the size of the subset to 290 solutions. Concerning Q_A, it is quickly decreased before the first 200 solutions, then it progresses slowly towards a constant value. It comes $Q_C = 303$ from Eq. 3 with $\Delta m = 20$. From here, increasing Q_C does not benefit diversity in the solution space. Hence, the stopping criterion for the exploration is fixed and represented in the objective space in Fig. 9a. Fig. 9b illustrates that increasing the number of solutions to improve the quality of the result may not benefit its analysis. It takes 3.02 seconds to compute those 303 solutions. The result of the exploration is noted Σ_{ice} thereafter.

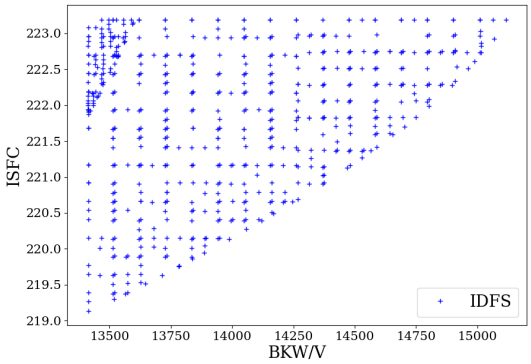
4.2.2 Results

Fig. 10 presents the objective space of the *ice* problem. The approximated Pareto front Θ_{ice} obtained with NSGA-II is composed of 190 design solutions (green triangles). The partial exploration is prematurely stopped after 303 solutions computed (blue crosses). The red dots give the extreme values of the Pareto front obtained from the single-objective optimization of each criterion with BARON.

The sole Pareto front presented in the introduction (Fig. 2a) is not sufficient to picture the full solution space and many quasi-optimal solutions are missed. The subset obtained with IDFS Σ_{ice} permits this analysis. The solutions from the partial exploration are represented in the design space in Fig. 11 using parallel coordinate plots. The two best designs regarding each objective are represented with dashed bold lines, their design variables also form the hull of the feasible space in Fig. 2a. Partial exploration reveals new feasible value combinations among the design variables. In particular, the available range value of c_r, d_E and w, respectively $[9.45, 9.74], [28.61, 30.99]$ and $[5.61, 6.5]$, are widened to $[9.3, 9.74], [28.61, 32.15]$ and $[5.25, 6.5]$. The exploration enabled to increase the volume of the design space by 320%. Indeed, NSGA-II guarantees diversity in the objective



(a) Partial exploration subset obtained with 303 solutions using the anytime BP algorithm IDFS



(b) Partial exploration subset obtained with 620 solutions ($Q_H \geq 90\%$) using the anytime B&P algorithm IDFS

Fig. 9: Representation of the objective space exploration at 2 different steps with IDFS

space but not in the design space when an anytime B&P algorithm is able to diversify in both the design space and the objective space. The evaluation of the overall spread Q_H in the design space and the hypervolume H_V in the objective space highlights this point. Fig. 12a represents the evolution of Q_H during the exploration computed on the design variables only. IDFS is able to quickly diversify the design parameter values, especially at the frontier of the search space. The evaluation of the hypervolume of Σ_{ice} is given in Fig. 12b. For this convex bi-objective problem, the capacity of computing the frontier enables to cover the Pareto front.

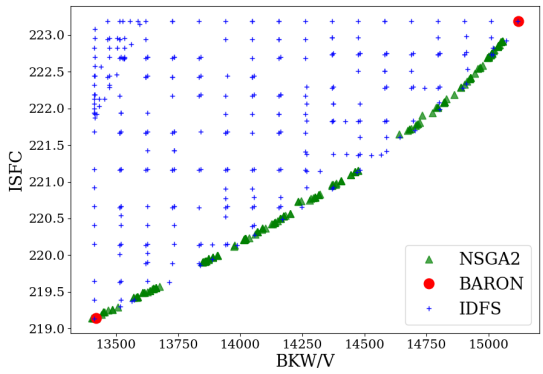


Fig. 10: Solutions space of the bi-objective *ice* problem (303 solutions with IDFS (blue crosses), 190 with NSGA-II (green triangles)

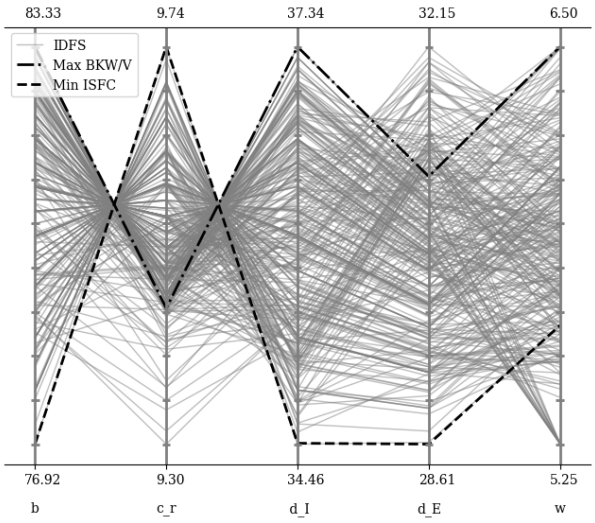
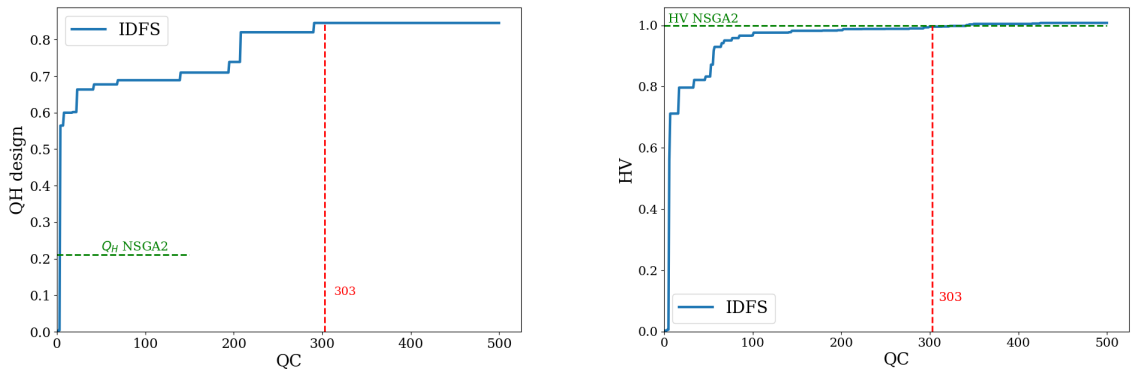


Fig. 11: Representation of 303 solutions from IDFS in the design variable space of the *ice* problem.

It is possible to go further in the experiment by tightening the search space closer to the Pareto front with the right ε-constraint. For this particular problem, the Pareto front can be approximated as a smooth curve using a log-linear regression of the second order. In a second phase, this relation is converted into a constraint c_{10} which represents a tolerance of 0.5% on the fuel consumption objective (see Eq. 5). This example illustrates the interaction between the user and an anytime B&P



(a) Evolution of Q_H evaluated on the design variables for IDFS (*ice* problem). The red dashed line gives the value of Q_H design for Θ_{ice}.

(b) Evolution of H_V with IDFS (*ice* problem). H_V is normalized with the H_V value of Θ_{ice} (reference point: $(13410, 223.2)$)

Fig. 12: Evolution of Q_H and H_V for both strategies.

algorithm solver to focus the exploration on a particular part of the search space. The results of this second exploration Σ^2_{ice} in both the objective and design spaces are given in Figures 13a and 13b. New designs are identified, i.e. their design variable values are out of the region Θ_{ice}. This second exploration obtained a design space 148% wider than the one obtained with a metaheuristic.

$$c_{10} \ : \ ISFC \leq f(BKW/V) \tag{4}$$

$$f(x) = 125.7ln(x)^2 - 2372.24ln(x) + 11412 \tag{5}$$

4.3 Actuator problem

4.3.1 Model

The analytical model of the electromechanical actuator problem [7] is built from simplified electromagnetic relations, following the conservation of the magnetic and thermodynamic flux. Table 3 presents the design variables and design criteria, including P the number of pole pairs as a discrete variable. The resulting MOP is a mixed-integer one with 9 inequality constraints (see Annex A.2).

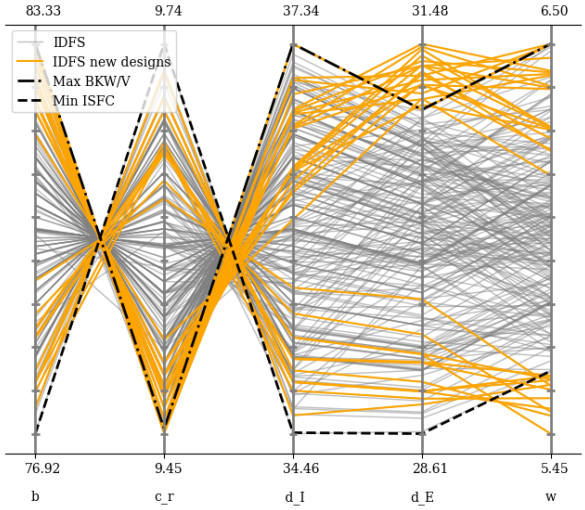
The evolutionary algorithm NSGA-III is used for the optimization approach due to its effectiveness in solving MOPs with 3 or more objectives [40]. The resolution is done with a population size of 156, 153 reference directions, and 1000 generations. Other computation parameters are presented in Table 2. The resulting approximation Θ_{act} obtained after 60 seconds contains 59 well-distributed solutions.

Table 3: Actuator main model parameters

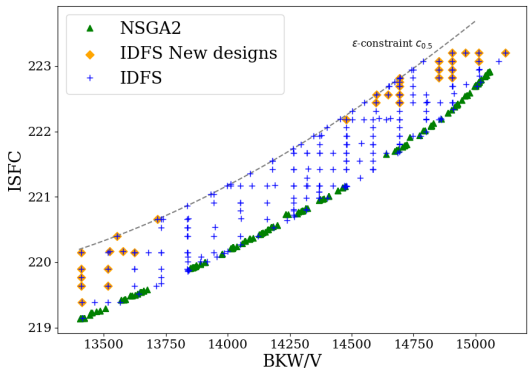
Name	Description	Variable type
e	Mechanical air gap (m)	Design
J_{cu}	Current areal density (A/m2)	Design
la	Thickness of the magnet (m)	Design
β	Polar arc coefficient	Design
P	Number of pole pairs	Design
Vu	volume of the active part (m3)	Objective
Va	volume of the magnet (m3)	Objective
Pj	Losses by joule effect (Watt)	Objective

The corresponding CSP to explore is built adding three inequality constraints on the objective variables (6). Those thresholds are given arbitrary, based on the data from the original paper. The same protocol as the one used for the *ice* problem is applied to size the result of the exploration with IDFS. A preliminary exploration stops after 2000 computed solutions, which enables to evaluate the evolution of the quality of the result with Q_H and Q_A. A compromise between diversity in the solution space and the size of the subset is established at 412 solutions (Σ_{act} computed after 90 seconds).

$$Vu \leq 6.5e-4 \ ; \ Va \leq 1.5e-4 \ ; \ Pj \leq 45 \tag{6}$$



(a) Parallel plot coordinate of the second partial exploration Σ^2_{ice} in the design space (*ice* problem + c_{11} constraint).



(b) Objective space of the *ice* problem. The exploration with IDFS is restricted with the added constraint c_{11}.

Fig. 13: Representation of solutions after adding a new ε-constraint to enclose the PF

4.3.2 Results

Fig. 14 shows the parallel plot coordinates of the designs from Σ_{act}. The values are normalized using Eq. 7 and the extreme values of Θ_{act} obtained with NSGA-III (Table 4). The variable P is not represented because the feasible space presents the unique value $P = 4$. In the design space $(\tilde{e}, \tilde{J_{cu}}, \tilde{la}, \tilde{\beta})$ the solutions are well distributed and IDFS is able to find many solutions out of the space covered by the result of NSGA-III (values greater than 1 or smaller than 0). Precisely, among the 412 solutions, 16 are in the PF design scope. This result is also visible in the evaluation of Q_H on the design space (Table 5) since we have $Q_H(\Sigma_{act}) \gg Q_H(\Theta_{act})$. The partial exploration of the *act* problem search space gives a wide representative sample of the feasible designs. The corresponding objective values are represented on the same graph $(\tilde{V_u}, \tilde{V_a}, \tilde{P_j})$.

$$\tilde{X} = \frac{X - X_{PF}^{min}}{X_{PF}^{max} - X_{PF}^{min}} \tag{7}$$

IDFS is not able to completely cover the range of the approximated Pareto front on V_u and P_j. Indeed, the minimum value previously obtained with NSGA-III for those two objectives are not reached by IDFS (corresponding to $\tilde{V_u} = 0$ and $\tilde{P_j} = 0$). It comes from the difficulty of the contracting operator to deal with the non-linearity of the active constraints in this area of the search space. The capacity of an anytime B& P algorithm to reach the Pareto front during the exploration can be evaluated through the hypervolume indicator H_V. Fig. 15 illustrates this result comparing H_V during the exploration with the reference value H_V(NSGA-III). The partial exploration slightly improves the minimum ideal value of the objective V_a ($\underline{\tilde{V_a}} = -0.0018$ in Fig. 14).

Eventually, IDFS finds feasible designs close to the optimum that are not identified by NSGA-III. The decision maker may consider those solutions for further investigation using additional criteria that are not implemented in the CSP model. Fig. 16 represents 48 of the close-to-optimum designs that are out of the scope of NSGA-III. Considering the hull of those 48 solutions, the feasible domain of the allowed mechanical air gap e is raised by more than 550% and the feasible domain of the current areal density J_{cu} is augmented by 80%.

Table 4: Bounds of the Pareto front computed with NSGA-III for the *act* problem.

Variable	min	max
e	1.00e-05	1.613e-05
Jcu	5.814e+06	7.208e+06
la	4.570-03	8.952e-03
beta	0.8	0.9967
p	4	4
Vu	5.337e-04	6.046e-04
Va	9.148e-05	1.377e-04
Pj	37.47	45

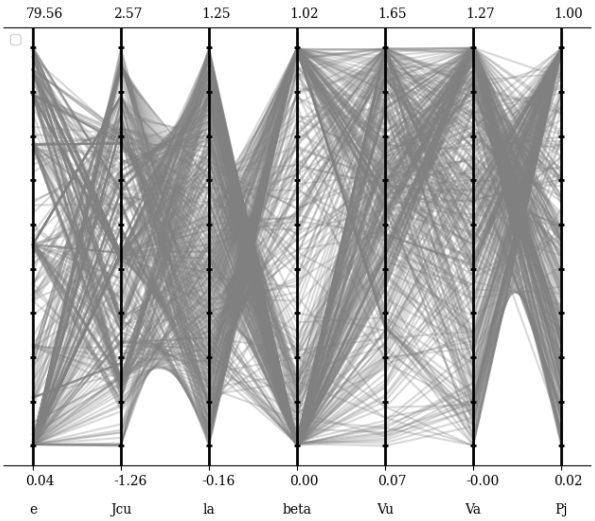


Fig. 14: Solutions obtained with IDFS, normalized with NSGA-III results (*act* problem).

Table 5: Comparison of the overall spread Q_H for the *act* problem.

	IDFS	NSGA-II
Q_H design	**0.2917**	0.0007
Q_H objective	**0.0568**	0.0358

4.4 Water problem

4.4.1 Model

The *water* problem is a five-objective optimization problem solved in [40]. This problem is composed of three continuous variables (x_1, x_2, x_3) and seven inequality constraints (aside variable domains restrictions). It permits to scale up our exploration approach. The resolution conditions used in [40] are reproduced for the optimization part of the experiments. NSGA-III is used with 212 individuals in the genetic population

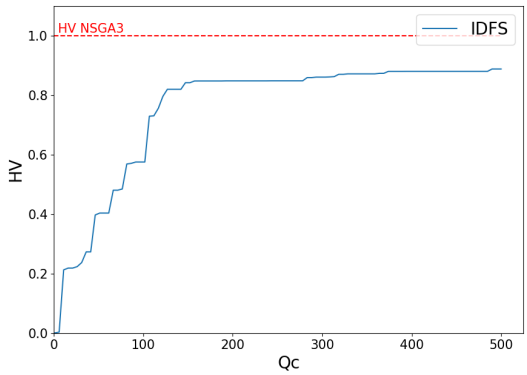


Fig. 15: Hypervolume evolution during the exploration of the search space of the *act* problem with IDFS algorithm

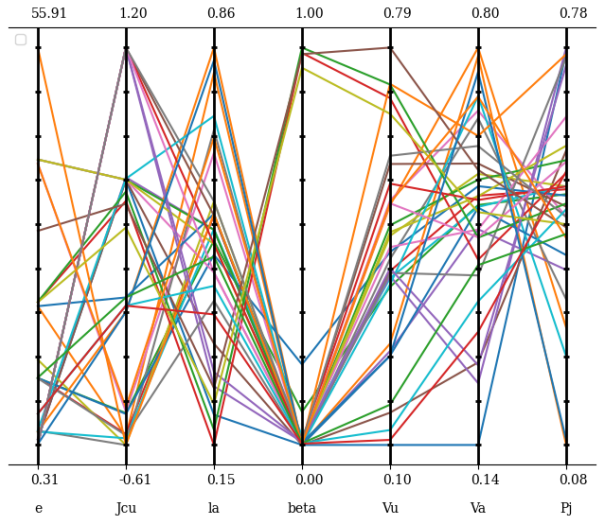


Fig. 16: New close-to-optimal designs obtained with IDFS (*act* problem).

and 210 reference points. After 1000 generations, this solver returns a hundred distributed solutions (Θ_{wat}) to approximate the Pareto front.

The equivalent CSP for the partial exploration is built with added inequality constraints on the objective parameters (Equation 8). Maximum thresholds are given arbitrary, without prior knowledge of the Pareto front. Using the IDFS strategy, 2000 solutions are first generated to study the quality of the partial exploration. A good compromise between the number of solutions and their distribution in the solution space

is established to 423 solutions based on the computation of Q_H and Q_A. Those 423 solutions are considered representative of the feasible space and noted Σ_{wat}. They are computed in 0.076 seconds.

$$f_1 \leq 8e4 \; ; \; f_2 \leq 1.35e3 \; ; f_3 \leq 1 \; ; \\ f_4 \leq 8.1e6 \; ; \; f_5 \leq 3e4 \tag{8}$$

4.4.2 Results

The 423 feasible designs are represented in both the design space (x_1, x_2, x_3) and the objective space $(f_1, f_2, f_3, f_4, f_5)$ in Fig. 17. The values are normalized using the extreme values of the approximated Pareto front (Eq. 7). Table 6 compares Q_H between the subsets Σ_{wat} and Θ_{wat}. IDFS conducts to a more diverse set. It covers a wider part of the design space, in particular on the variable x_3 (corresponding to values greater than 1). IDFS have difficulties to converge close to the expected minima on the objectives f_1, f_2 and f_5, hence, the partial exploration does not completely cover the Pareto front. This point is clearly illustrated in Fig. 18, and we see that IDFS does not reach the H_V value of the Pareto front. Among those 423 solutions, some are new designs close to the optimum, which may be interesting to consider. In Fig. 19, filters are applied to represent those solutions with $\tilde{x_3} \geq 1$ and $\tilde{f}_i \leq 0.8$.

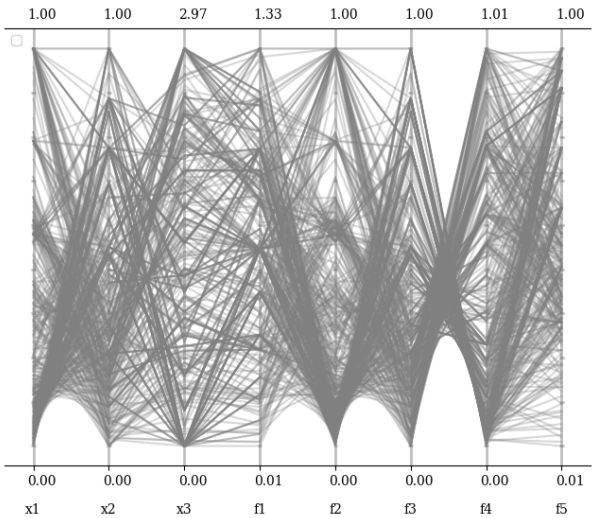


Fig. 17: Normalized solutions obtained with IDFS (*wat* problem).

Table 6: *wat* problem. Comparison of the overall spread Q_H .

	IDFS	NSGA-II
QH design	**0.99**	0.337
QH objective	**0.78**	0.62

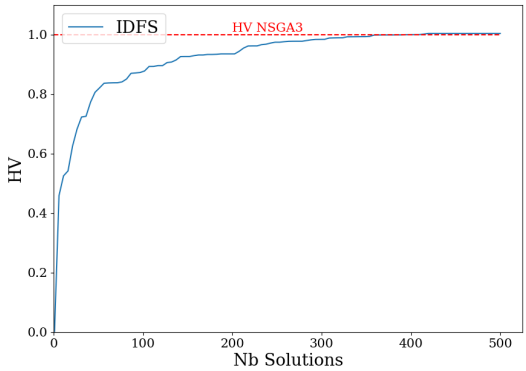


Fig. 18: Hypervolume evolution during the exploration of the search space of the *wat* problem with IDFS algorithm

4.5 Summary of results and discussion

The first part of the experiments compares the anytime exploration method with the multi-objective optimization approach on a simple internal combustion engine problem with two design criteria. While NSGA-II computes a clear bi-objective Pareto front, our anytime B&P algorithm returns a good representation of the full solution space. The designs composing the subset are diversified on both the objective space and the design space, which permits to identify new near-optimal designs. In a second phase, an interactive process using a partial exploration solver allows to identify new close-to-optimal designs. Indeed, once a first representative set of solutions is computed, it is possible to focus the search on a particular part of the feasible space, adding specific constraints. Our algorithm is also applied on more complex problems, *act* and *wat*, with respectively three and five design criteria. IDFS is here compared with NSGA-III. In both case studies, NSGA-III computes at most a hundred solutions well distributed in the objective space.

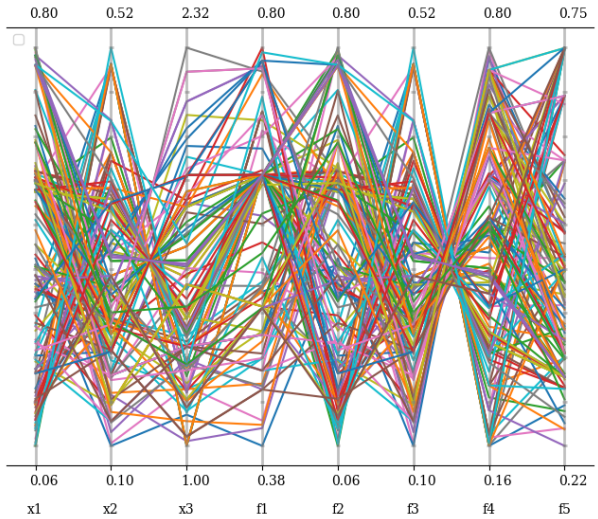


Fig. 19: New designs obtained with IDFS in Σ_{wat} (*wat* problem).

With IDFS, we are able to compute representative sets of solutions composed of a few hundred solutions with a high diversity in both the design space and the objective space. It shows that partial exploration is able to both reach the objective space close to the Pareto Front and widen the feasible domains in the design space.
Partial design space exploration, as proposed in this paper, presents the benefit to get a better understanding of the whole feasible space, prior to any optimization consideration. So-called design objectives are easily considered through ε-constraints. The complexity of a partial exploration with an anytime B&P solver is also independent on the number of objectives considered. On the contrary, optimization gets harder with the growing number of objectives [21, 22]. This advantage may particularly be interesting in eco-design where several environmental impacts can be considered as distinct design criteria.

Furthermore, the models studied here are coarse-grained models, built for preliminary design to reduce quickly the feasible space before more in depth process. Those models are simplifications of reality based on important hypotheses. Hence, optima obtained from MOP solvers may not be the best designs to consider in reality. For example, the internal combustion engine model uses an ideal thermal efficiency air-cycle [14] and the electromechanical actuator considers simplified modeling such as the conservation of the flux in the magnetic circuit [7]. Partial exploration permits to loosen the rigor coming with optimization methods on imperfect models.

Eventually, the interactive process presented on the *ice* problem illustrates the potential of this framework to integrate interactivity in the design process. Indeed, the process based on anytime B&P algorithm supports *stop and start* mechanisms such that the resolution can be resumed from any point of the exploration.

5 Conclusion

CSP modeling represents an important tool for design in preliminary phases to explore feasible solutions. This work proposes to face preliminary design problems using partial design space exploration and to compare it with classical optimization methods. For this purpose an anytime B&P algorithm (IDFS) is implemented with a specific search strategy. The algorithm is able to tackle CSP with mixed integer variables and non-linear constraints. It seeks to return a subset of solutions diverse enough to give a good representation of the whole feasible space.

Our algorithm is evaluated on three different design model. Their design criteria are converted into thresholds for the exploration with IDFS. Two quality indicators Q_A and Q_H are introduced to evaluate its capacity to return wide and uniform subsets of solutions. The hypervolume H_V is also used for the analysis and the comparison with more classical optimization approaches based on metaheuristics. It appears that IDFS is able to return representative subsets composed of a few hundreds of solutions in a reasonable time. Those designs are diversified in the objective space, but also in the design space. Finally, it permits to identify interesting close-to-optimal designs which were not generated by the metaheuristics.

The discussion brings out the interest of the partial exploration framework according to our experiments. This anytime exploration paradigm enables to diminish the black-box aspect of a decision-making program, giving users the opportunity to easily iterate between a model and its solution space. A representative partial set of solutions permits to picture the feasible space and considers well diversified designs. The next

step of this framework is to built an incremental and interactive methodology to both generate and analyse feasible architectures in a preliminary design process.

5.1 Statements and Declarations

The authors have no competing interest to declare.

References

[1] Tchertchian N, Yvars P, Millet D. Benefits and Limits of a Constraint Satisfaction Problem/Life Cycle Assessment Approach for the Ecodesign of Complex Systems: A Case Applied to a Hybrid Passenger Ferry. Journal of Cleaner Production. 2013;.

[2] Kang E, Jackson E, Schulte W. An Approach for Effective Design Space Exploration. In: Calinescu R, Jackson E, editors. Found. Comput. Softw. Model. Dev. Verification Adapt. Syst. Lecture Notes in Computer Science. Berlin, Heidelberg: Springer; 2011. p. 33–54.

[3] Gelle E, Faltings B, Clément DE, Smith I. Constraint Satisfaction Methods for Applications in Engineering. Eng Comput. 2000 Sep;16. https://doi.org/10.1007/PL00007190.

[4] Yvars PA. Using Constraint Satisfaction for Designing Mechanical Systems. Int J Interact Des Manuf IJIDeM. 2008 Jan;2:161–167. https://doi.org/10.1007/s12008-008-0047-3.

[5] Larroude V, Yvars PA, Millet D. Global Optimization of Environmental Impact by a Constraint Satisfaction Approach - Application to Ship-Ecodesign. Des Soc. 2011;p. 11.

[6] Cicconi P, Manieri S, Nardelli M, Bergantino N, Raffaeli R, Germani M. A Constraint-Based Approach for Optimizing the Design of Overhead Lines. Int J Interact Des Manuf. 2020 Sep;14(3):1121–1139. https://doi.org/10.1007/s12008-020-00680-x.

[7] Messine F, Nogarede B, Lagouanelle JL. Optimal Design of Electromechanical Actuators: A New Method Based on Global Optimization. IEEE Trans Magn. 1998 Jan;34(1):299–308. https://doi.org/10.1109/20.650361.

[8] Yvars PA, Lafon P, Zimmer L. Optimization of Mechanical System: Contribution of Constraint Satisfaction Method. In: 2009 Int. Conf. Comput. Ind. Eng. Troyes, France: IEEE; 2009. p. 1379–1384.

[9] Brownlee AEI, Wright JA. Constrained, Mixed-Integer and Multi-Objective Optimisation of Building Designs by NSGA-II with Fitness Approximation. Applied Soft Computing. 2015 Aug;33:114–126. https://doi.org/10.1016/j.asoc.2015.04.010.

[10] Demarco F, Bertacchini F, Scuro C, Bilotta E, Pantano P. The Development and Application of an Optimization Tool in Industrial Design. Int J Interact Des Manuf. 2020 Sep;14(3):955–970. https://doi.org/10.1007/s12008-020-00679-4.

[11] Fortunet C, Durieux S, Chanal H, Duc E. Multicriteria Decision Optimization for the Design and Manufacture of Structural Aircraft Parts. Int J Interact Des Manuf. 2020 Sep;14(3):1015–1030. https://doi.org/10.1007/s12008-020-00685-6.

[12] Stewart RH, Palmer TS, DuPont B. A Survey of Multi-Objective Optimization Methods and Their Applications for Nuclear Scientists and Engineers. Progress in Nuclear Energy. 2021 Aug;138:103830. https://doi.org/10.1016/j.pnucene.2021.103830.

[13] Zavala GR, Nebro AJ, Luna F, Coello Coello CA. A Survey of Multi-Objective Metaheuristics Applied to Structural Optimization. Struct Multidisc Optim. 2014 Apr;49(4):537–558. https://doi.org/10.1007/s00158-013-0996-4.

[14] Papalambros P, Wilde D. Principles of Optimal Design: Modeling and Computation. 3rd ed. Modelling and computation; 2017.

[15] Deb K, Agrawal S, Pratap A, Meyarivan T. A Fast Elitist Non-dominated Sorting Genetic Algorithm for Multi-objective Optimization: NSGA-II. In: Parallel Problem Solving from

Nature PPSN VI. vol. 1917. Berlin, Heidelberg: Springer Berlin Heidelberg; 2000. p. 849–858.

[16] Mandal WA. A Two-Bar Truss Structural Model under Uncertainty: A Uncertain Chance Constrained Geometric Programming (UCCGP) Approach. Int J Interact Des Manuf. 2019 Jun;13(2):471–485. https://doi.org/10.1007/s12008-018-0477-5.

[17] Institute for Environment and Sustainability (Joint Research Centre). Life Cycle Indicators Framework: Development of Life Cycle Based Macro Level Monitoring Indicators for Resources, Products and Waste for the EU 27. LU: Publications Office of the European Union; 2012.

[18] Deb K, Jain S. Running Performance Metrics for Evolutionary Multi-Objective Optimizations; 2002. .

[19] Schwind N, Okimoto T, Clement M, Inoue K. Representative Solutions for Multi-Objective Constraint Optimization Problems. In: The 15th International Conference on Principles of Knowledge Representation and Reasoning (KR'16). KR; 2016. .

[20] Ingmar L, de la Banda MG, Stuckey PJ, Tack G. Modelling Diversity of Solutions. Proc AAAI Conf Artif Intell. 2020 Apr;34(02):1528–1535. https://doi.org/10.1609/aaai.v34i02.5512.

[21] Vassilvitskii S, Yannakakis M. Efficiently Computing Succinct Trade-off Curves. In: Theor. Comput. Sci. 348 2005; 2005. p. 334–356.

[22] Yuan Y, Ong YS, Gupta A, Xu H. Objective Reduction in Many-Objective Optimization: Evolutionary Multiobjective Approaches and Comprehensive Analysis. IEEE Trans Evol Comput. 2018 Apr;22(2):189–210. https://doi.org/10.1109/TEVC.2017.2672668.

[23] Van Hentenryck P, Mcallester D, Kapur D. Solving Polynomial Systems Using a Branch and Prune Approach. SIAM J Numer Anal. 1997;34. https://doi.org/10.1137/S0036142995281504.

[24] Chenouard R, Goldsztejn A, Jermann C. Search Strategies for an Anytime Usage of the Branch and Prune Algorithm. In: Int. Jt. Conf. Artif. Intell. United States; 2009. p. p. 468–473.

[25] Benhamou F, Goualard F, Granvilliers L, Puget JF. Revising Hull and Box Consistency. Int Conf Log Program. 1999;p. 230–244. https://doi.org/10.7551/mitpress/4304.003.0024.

[26] Araya I, Trombettoni G, Neveu B. A Contractor Based on Convex Interval Taylor. In: Integr. AI Tech. Contraint Program. Comb. Optimzation Probl. Lecture Notes in Computer Science. Berlin, Heidelberg: Springer; 2012. p. 1–16.

[27] Chabert G, Jaulin L. Contractor Programming. Artif Intell. 2009;173(11):1079–1100. https://doi.org/10.1016/j.artint.2009.03.002.

[28] Ryoo HS, Sahinidis NV. A Branch-and-Reduce Approach to Global Optimization. J Glob Optim. 1996 Mar;8(2):107–138. https://doi.org/10.1007/BF00138689.

[29] Misener R, Floudas CA. ANTIGONE: Algorithms for coNTinuous / Integer Global Optimization of Nonlinear Equations. J Glob Optim. 2014 Jul;59(2):503–526. https://doi.org/10.1007/s10898-014-0166-2.

[30] Ehrgott M, Ruzika S. Improved ϵ-Constraint Method for Multiobjective Programming. J Optim Theory Appl. 2008 Sep;138(3):375–396. https://doi.org/10.1007/s10957-008-9394-2.

[31] Audet C, Bigeon J, Cartier D, Le Digabel S, Salomon L. Performance Indicators in Multiobjective Optimization. European Journal of Operational Research. 2020 Nov;292. https://doi.org/10.1016/j.ejor.2020.11.016.

[32] Zitzler E, Thiele L, Laumanns M, Fonseca CM, Fonseca V. Performance Assessment of Multiobjective Optimizers: An Analysis and Review. IEEE Trans Evol Comput.

2003 May;7:117–132. https://doi.org/10.1109/TEVC.2003.810758.

[33] Palesi M, Givargis T. Multi-Objective Design Space Exploration Using Genetic Algorithms. In: Hardware/Software Codesign - Proceedings of the International Workshop; 2002. p. 67–72.

[34] Cagan J, Campbell M, Finger S, Tomiyama T. A Framework for Computational Design Synthesis: Model and Applications. J Comput Inf Sci Eng - JCISE. 2005 Sep;5. https://doi.org/10.1115/1.2013289.

[35] Chenouard R, Granvilliers L, Sebastian P. Search Heuristics for Constraint-Aided Embodiment Design. AIEDAM. 2009 May;23(2):175–195. https://doi.org/10.1017/S0890060409000055.

[36] Sayın S. Measuring the Quality of Discrete Representations of Efficient Sets in Multiple Objective Mathematical Programming. Math Program. 2000 May;87(3):543–560. https://doi.org/10.1007/s101070050011.

[37] Dechter R, Pearl J. Generalized Best-First Search Strategies and the Optimality of A*. J ACM. 1985;32(3). https://doi.org/10.1145/3828.3830.

[38] Morrison DR, Jacobson SH, Sauppe JJ, Sewell EC. Branch-and-Bound Algorithms: A Survey of Recent Advances in Searching, Branching, and Pruning. Discrete Optim. 2016;19:79–102. https://doi.org/10.1016/j.disopt.2016.01.005.

[39] Meseguer P. Interleaved Depth-First Search. Proc Fifteenth Int Jt Conf Artifical Intell - Vol 2. 1997;p. 1382–1387. https://doi.org/10.5555/1622270.1622355.

[40] Jain H, Deb K. An Evolutionary Many-Objective Optimization Algorithm Using Reference-Point Based Nondominated Sorting Approach, Part II: Handling Constraints and Extending to an Adaptive Approach. IEEE Trans Evol Comput. 2014 Aug;18(4):602–622. https://doi.org/10.1109/TEVC.2013.2281534.

Appendix A Models

A.1 Engine

$$
\begin{aligned}
CSP: \ \mathbf{x} =& (b, c_r, d_I, d_E, w)\\
[b] =& [0, 100]\\
[c_r] =& [0, 100]\\
[d_I] =& [0, 0.05]\\
[d_E] =& [0, 1]\\
[w] =& [0, 6.5]
\end{aligned}
$$

subject to

$$
\begin{aligned}
c_1 &: b - 83.33 \leq 0\\
c_2 &: 76.9 - b \leq 0\\
c_3 &: d_I + d_E - 0.82b \leq 0\\
c_4 &: 0.83d_I - d_E \leq 0\\
c_5 &: d_E - 0.89d_I \leq 0\\
c_6 &: 215.46 - d_I^2 \leq 0\\
c_7 &: c_r - 13.2 + 0.0045b \leq 0\\
c_8 &: 2b^3/1.18348e^6 - 1.3 < 0\\
c_9 &: 2b^3/1.18348e^6 - 0.7 > 0
\end{aligned}
$$

$$
\begin{aligned}
MOP: & \\
& min(ISFC, -BKW/V)\\
ISFC =& 81.8964/(0.8595(1 - c_r^{-0.33})\\
& + (0.83((8 + 4c_r) + 1.5(c_r\\
& - 1)(4\pi/1.859e^6)b^3)/(2 + c_r)b))\\
BKW/V =& w[FMEP - 3688\eta_t\eta_v]/120
\end{aligned}
$$

where

$$
\begin{aligned}
FMEP =& 4.826(c_r - 9.2) + (7.97 + 2.99e^5wb^{-2}\\
& + 1.36e7bw^2)\\
\eta_t =& 0.8595(1 - c_r^{-0.33}) - S_v\sqrt{1.5/w}\\
\eta_v =& \eta_{vb}(1 + 5.96e^{-3}w^2)/((1 + 55.789/0.44)\\
& \times (w/(d_I)^2))^2\\
\eta_{vb} =& \begin{cases} 0.067 - 0.038e^{w-5.25}, & w \geq 5.25,\\ 0.637 + 0.13w - 0.014w^2 & \\ +0.00066w^3, w < 5.25 & \end{cases}\\
S_v =& 0.83(8 + 4c_r + 1.0134e^{-5}(c_r - 1)b^3)/((2 + c_r)b)
\end{aligned}
$$

$$
\begin{aligned}
Exploration: & \\
& BWK \geq 13400\\
& ISFC \leq 223.2
\end{aligned}
$$

A.2 Actuator

$$
\begin{aligned}
&CSP: \\
&\quad \mathbf{x} = (e, J_{cu}, la, \beta, P) \\
&\quad [e] = [0.00001, 0.005] \\
&\quad [J_{cu}] = [1e5, 1e7] \\
&\quad [la] = [0.001, 0.05] \\
&\quad [\beta] = [0.8, 1] \\
&\quad [P] = [1, 10] \\
&\textit{subject to.} \\
&\quad c_1 : D - 2 * (la + e) \geq 0 \\
&\quad c_2 : 0.001 \leq E \leq 0.05 \\
&\quad c_3 : 0.001 \leq D \leq 0.05 \\
&\quad c_4 : 0.001 \leq \lambda \leq 0.05 \\
&\quad c_5 : 0.001 \leq K_f \leq 0.05 \\
&\textit{where} \\
&\quad E = 10^{11}/(0.7 * (J_{cu}^2)) \\
&\quad D = 0.1P/\pi \\
&\quad \lambda = 0.015\pi D^2 (D + E) B_e \sqrt{7^{11}\beta E} \\
&\quad Kf = 1.5P\beta(e + E)/D \\
&\quad B_e = (1.8la/(D \log(D + 2E)/(D - 2(la + e))) \\
&\quad C = D\pi\beta B_e/(6P) \\
&MOP: \\
&\quad min(Vu, Va, Pj) \\
&\quad Vu = \pi(D/\lambda)(D + E - e - la)(2C + E + e + la) \\
&\quad Va = \pi\beta la(D/\lambda)(D - 2e - la) \\
&\quad Pj = 0.018e^{-6}\pi(D/\lambda)(D + E)10^{11} \\
&\textit{Exploration} \\
&\quad Vu \leq 6.5e^{-4} \\
&\quad Va \leq 1.5e^{-4} \\
&\quad Pj \leq 45
\end{aligned}
$$

A.3 Water

$$
\begin{aligned}
&CSP: \\
&\quad \mathbf{x} = (x_1, x_2, x_3) \\
&\quad [x_1] = [0.01, 0.45] \\
&\quad [x_2] = [0.01, 0.10] \\
&\quad [x_3] = [0.01, 0.10] \\
&\textit{subject to.} \\
&\quad c_1 : 0.00139/(x_1x_2) + 4.94x_3 - 0.08 \leq 1 \\
&\quad c_2 : 0.000306/(x_1x_2) + 1.082x_3 - 0.0986 \leq 1 \\
&\quad c_3 : 12.307/(x_1x_2) + 49408.24x_3 \\
&\qquad + 4051.02 \leq 50000 \\
&\quad c_4 : 2.098/(x_1x_2) + 8046.33x_3 - 696.71 \leq 16000 \\
&\quad c_5 : 2.138/(x_1x_2) + 7883.39x_3 - 705.04 \leq 10000 \\
&\quad c_6 : 0.417/(x_1x_2) + 1721.26x_3 - 136.54 \leq 2000 \\
&\quad c_7 : 0.164/(x_1x_2) + 631.13x_3 - 54.48 \leq 550 \\
&MOP: \\
&\quad min(f_1, f_2, f_3, f_4, f_5) \\
&\quad f_1 = 106780.37(x_2 + x_3) + 61704.67 \\
&\quad f_2 = 3000x_1 \\
&\quad f_3 = 305700.02289x_2/(0.062289.0)0.65 \\
&\quad f_4 = 250.02289.0exp(-39.75x_2 + 9.9x_3 + 2.74) \\
&\quad f_5 = 25.0((1.39/(x_1x_2)) + 4940.0x_3 - 80.0) \\
&\textit{Exploration} \\
&\quad f_1 \leq 8e4 \\
&\quad f_2 \leq 1.35e3 \\
&\quad f_3 \leq 1 \\
&\quad f_4 \leq 8.1e6 \\
&\quad f_5 \leq 3e4
\end{aligned}
$$

Highlights and graphical abstract

- Partial exploration aims to compute a diversified subset of feasible solutions;
- We built an anytime branch and prune algorithm for partial design space exploration
- We built a protocol to analyze diversity in both the design and the objective space
- We compare partial exploration and optimization approaches on three design problems
- Partial Exploration is a tool for decision makers to identify quasi-optimal designs

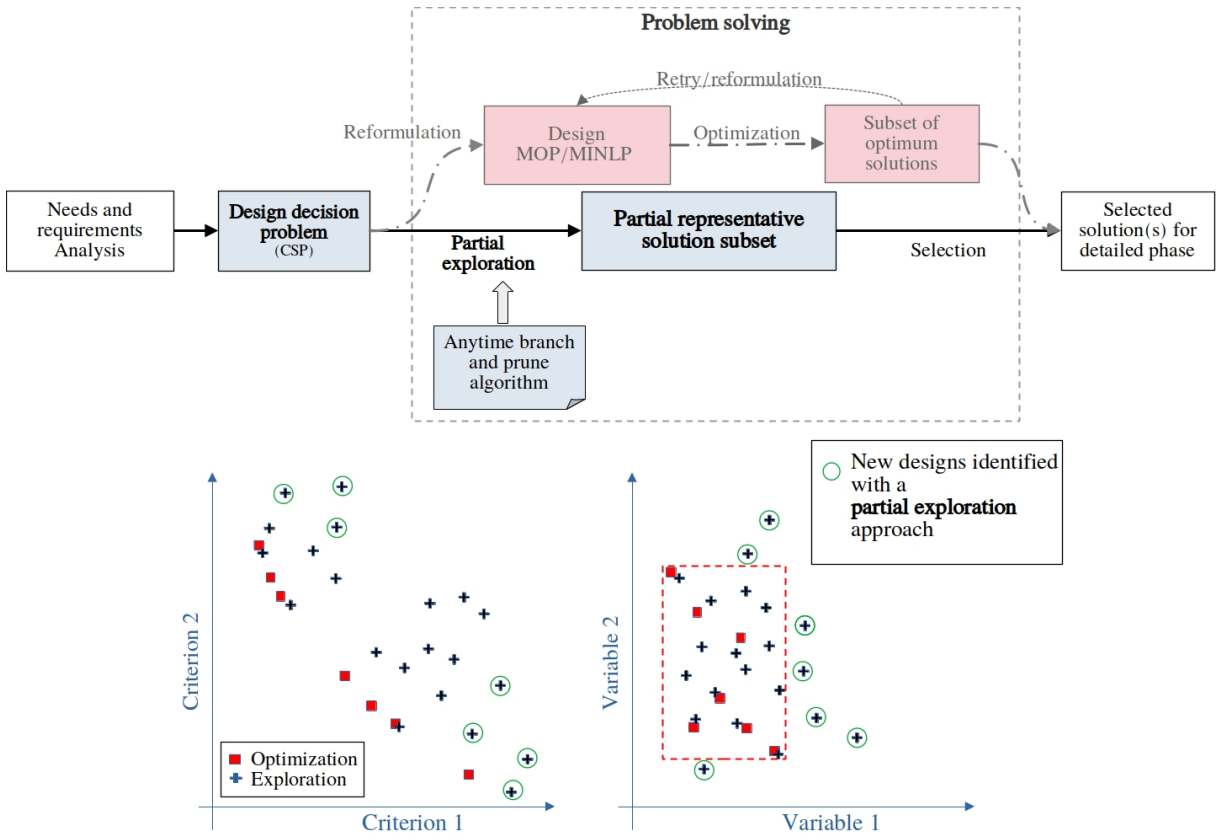


Fig.a - Objective space

Fig.b - Design space

Fig. A1: Graphical Abstract