Noname manuscript No.
(will be inserted by the editor)

Domain decomposition based parallel Howard's algorithm

Adriano Festa

Received: date / Accepted: date

Abstract The Classic Howard's algorithm, a technique of resolution for discrete Hamilton-Jacobi equations, is of large use in applications for its high efficiency and good performances. A special beneficial characteristic of the method is the superlinear convergence which, in presence of a finite number of controls, becomes in finite time. These features give to the method a special interest in the resolution of problems with a big dimensionality, common issue in the sector. Performances of the method can be significantly improved by using parallel calculus; how to build a parallel version of method is not a trivial point, the difficulties come from the strict relation between various values of the solution, even related to distant points of the domain. In this contribution we propose an original parallel version of the Howard's algorithm driven by an idea of domain decomposition. This permit to derive some important properties and proving the convergence under quite standard assumptions. The key point about the conditions of connection between the sub domains will be discussed and validated. Through some tests and examples will be shown the good features of the algorithm.

Keywords Howard's algorithm (policy iterations), Semi-smooth Newton's method, Parallel Calculus, Domain Decomposition

Mathematics Subject Classification (2000) 49M15 · 65Y05 · 65N55

This work was supported by the European Union under the 7th Framework Programme FP7-PEOPLE-2010-ITN SADCO, Sensitivity Analysis for Deterministic Controller Design.

Adriano Festa
ENSTA ParisTech
Tel.: +33 (0) 782271537
E-mail: adriano.festa@ensta.fr
91120 Palaiseau

1 Introduction

The Howard's algorithm (also called policy iteration algorithm) is a classical method for solving a discrete Hamilton-Jacobi equation. This technique, developed by Bellman and Howard [5,13], is of large use in applications, thanks to its good proprieties of efficiency and simplicity. This method, using the concept of *slant differentiability* introduced in [15],[16], can be shown to be of semi-smooth Newton's type, with all the good features in term of superlinear convergence and, in some cases of interest, even quadratic convergence.

In this paper, we propose a parallel version of it, discussing the advantages and the weak points of such proposal.

To validate the steps necessary to the building of the technique, we will use a theoretical construction inspired by some recent results on domain decomposition (for example Barles, Briani and Chasseigne [2], or Rao and Zidani [17] based on the concept of Essential Hamiltonian [4]). Despite that, thanks to a greater regularity of the Hamiltonian, the decomposition can be studied just using standard techniques.

Parallel Calculus applied to Hamilton Jacobi equations resolution is a subject of actual interest because of the strict limitation of serial tools in real problems, where the memory storage restrictions and limits in the CPU speed, cause easily the infeasibility of the computation, even in cases relatively easy. With the purpose to build a parallel solver, the main problem to deal with is to manage the information passing through the threads. Our analysis is not the first contribution on the topic, but this is an original study of the specific possibilities offered by the Howard's algorithm.

At our knowledge the first parallel algorithm proposed was by Sun in 1993 [20] on the numerical solution of the Bellman equation related to an exit time problem for a diffusion process (i.e. for second order elliptic problems); and by Camilli, Falcone, Lanucara and Seghini [7] where an operator of the semiLagrangian kind, was proposed and studied on the interfaces of splitting. More recently, the issue was discussed also by Zhou and Zhan [22] where, passing to a quasi variational inequality formulation equivalent, there was possible a domain decomposition.

Our intention is to show a different way to approach to the topic. Decomposing the problem directly in its differential form, effectively, it is possible to give a coherent interpretation to the condition to impose on the boundaries of the sub-domains. Thereafter, passing to a discrete version of such decomposed problem it becomes relatively easy to show the convergence of the technique to the correct solution, avoiding the technical problems, elsewhere observed, about the manner to exchange information between the sub-domains. In our technique, as explained later, we will substitute it with the resolution of an auxiliary problem living in the interface of connection in the domain decomposition. In this way, data will be passed implicitly through the sub-problems.

The paper is structured as follows: in section 2 we recall the classic Howard's algorithm and the relation with the differential problem, focusing on the case of its Control Theory interpretation. In section 3, after mentioning the decomposition

results useful in the following, we present the algorithm, and we discuss the convergence. Section 4 is dedicated to a presentation of the performances and to show the advantages with respect the non parallel version. We will end present some possible extensions of the technique to some problems of interest: in the presence of a target to reach, obstacle constraints in the domain, max-min problems.

2 Classic Howard's algorithm

The problem considered is the following. Let be Ω open domain of $\mathbb{R}^d$ $(d \geq 1)$; the steady, first order, *Hamilton-Jacobi equation* (HJ) is described in the following form:

$$\begin{cases} \lambda v(x) + H(x, Dv(x)) = 0 & x \in \Omega \\ v(x) = g(x) & x \in \partial\Omega \end{cases} \tag{1}$$

where, following its *Optimal Control interpretation*, $\lambda \in \mathbb{R}^+$ is the *discount factor*, $g : \Omega \to \mathbb{R}$ is the *exit cost*, and the *Hamiltonian* $H : \Omega \times \mathbb{R}^d \to \mathbb{R}$ is defined by: $H(x,p) := \inf_{\alpha \in \mathcal{A}}\{-f(x,\alpha) \cdot p - l(x,\alpha)\}$ with $f : \Omega \times \mathcal{A} \to \mathbb{R}$ (*dynamics*) and $l : \Omega \times \mathcal{A} \to \mathbb{R}$ (*running cost*). This choice it is not restrictive but useful to simplify the presentation. As extension of the techniques we are going to present, it will be shown, in the dedicate section, as the same results can be obtained in presence of different kind of Hamiltonians, as in obstacle problems or in differential games.

Under classical assumptions on the data, it is known (see [1], [11]) that the equation (14) admits a unique solution $v : \overline{\Omega} \to \mathbb{R}$ in the weak sense of *viscosity solutions*. We can assume the global Lipschitz continuity of $f(\cdot,\alpha)$ and $l(\cdot,\alpha)$ with the last one uniformly bounded and the compactness of $\mathcal{A}$; these conditions are sufficient to prove the existence and the uniqueness of a solution in the viscosity sense.

The solution v is the value function to the infinite horizon problem with exit cost, where τ_x is the *first time of exit* form Ω:

$$\begin{cases} v(x) = \inf\limits_{a(\cdot) \in L^\infty([0,+\infty[;\mathcal{A})} \displaystyle\int_0^{\tau_x(a)} l(y_x(s), a(s)) e^{-\lambda s}\, ds + e^{-\lambda \tau_x(a)} g(y_x(\tau_x(a))), \\ \text{where } y_x(\cdot) \text{ is a.e. solution of} \quad \begin{cases} \dot{y}(t) = f(y(t), a(t)) \\ y(0) = x \end{cases} \end{cases}$$

Numerical schemes for approximation of such problem have been proposed from the early steps of the theory, let us mention the classical Finite Differences Schemes [10], [19], semiLagrangian [12], Discontinuous Galerkin [9] and many others.

In this paper we will focus on a *monotone*, *consistent* and *stable* scheme (class including the first two mentioned above), which will provide us the discrete problem where to apply the Howard's Algorithm.

Considered a discrete grid G with N points x_j, $j = 1, ..., N$ on the domain $\overline{\Omega}$, the finite N-dimensional approximation of v, V, will be the solution of the following discrete equation $(V_j = V(x_j))$

$$V_i = F_i^h(V_1, ..., V_N), \qquad i \in \{1, ..., N\} \tag{2}$$

where $h := \max diam S_j$, (maximal diameter of the family of simplices S_j built on G) is the discretization step, and related to a subset of the V_j, there are included the Dirichlet conditions following the obvious pattern

$$F_j^h(V_1, ..., V_N) := g(x_j), \quad x_j \in \partial\Omega.$$

We will assume on F, some Hypotheses sufficient to ensure the convergence of the discretization

(H1) *Monotony.* For every choice of two vectors V, W such that, $V \geq W$ (componentwise) then $F_i^h(V_1, ..., V_N) \geq F_i^h(W_1, ..., W_N)$ for all $i \in \{1, ..., N\}$.

(H2) *Stability.* If the data of the problem are finite, for every vector V, there exists a $C \geq 0$ such that V, solution of (2), is bounded by C i.e. $\|V\|_\infty = \max_{i=1,...,N} |F_i^h(V_1, ..., V_N)| \leq C$ independently from h.

(H3) *Consistency.* $\varphi(y) - F_i^h(\varphi(y_1) + \xi, ..., \varphi(y_N) + \xi) \to \lambda\varphi(x) + H(x, \varphi(x), D\varphi(x))$ for $\varphi \in C^1(\Omega)$, $x \in \Omega$, with $h \to 0$, $y \to x$, and $\xi \to 0$.

Generally, numerical solution is archived by a fixed point iteration of the kind $V_i^{s+1} = F_i^h(V^s)$.

Under these assumptions it has been discussed and proved [19] that V, solution of (2), converges to v, viscosity solution of (14) for $h \to 0$.

The special form of the Hamiltonian H gives us a correspondent special structure of the scheme F, in particular, with a rearrangement of the terms, the discrete problem (2) can be written as a resolution of a nonlinear system in the following form:

$$\text{Find } V \in \mathbb{R}^N; \quad \min_{\alpha \in \mathcal{A}^N} (B(\alpha)V - c_g(\alpha)) = 0 \tag{3}$$

where B is a $N \times N$ matrix and c_g is a N vector. The name c_g is chosen just to underline (it will be important in the following) that in that vector there are contained the information about the Dirichlet conditions imposed on the boundary. The *Policy Iteration Algorithm* (or Howard's Algorithm) consists in a two-steps iteration with an alternating improvement of the policy and the value function, as shown in Table 1.

It is by now known [6] that under a monotonicity assumption on the matrices $B(\alpha)$, automatically derived from $H1$ (as shown below), the above algorithm is a non smooth Newton method that converges superlinearly to the discrete solution of problem. The convergence of the algorithm is also discussed in the earlier work [18, **?**] where the results are given in a more regular framework.

Additionally, if $\mathcal{A}$ has a finite number of elements, and this is the standard case of a discretized space of the controls, then the algorithm converges in a finite number of iterations.

Proposition 1 *Let us assume $H1$. Then if the matrix $B(\alpha)$ is invertible, then it is monotone and not null for every $\alpha \in \mathcal{A} \cap \arg\min B(\alpha)V - c_g(\alpha)$ for a $V \in \mathbb{R}^n$.*

Howard's Algorithm (HA)
Inputs: $B(\cdot)$, $c_g(\cdot)$. (Implicitly, the values of V at the boundary points) Initialize $V^0 \in \mathbb{R}^N$ and $\alpha_0 \in \mathcal{A}^N$ Iterate $k \geq 0$: i) Find $V^k \in \mathbb{R}^N$ solution of $B(\alpha^k)V^k = c_g(\alpha^k)$. If $k \geq 1$ and $V^k = V^{k-1}$, then stop. Otherwise go to (ii). ii) $\alpha^{k+1} := \arg\min_{\alpha \in \mathcal{A}^n} \big(B(\alpha)V^k - c_g(\alpha)\big)$. Set $k := k+1$ and go to (i) Outputs: V^{k+1}.

Table 1 Pseudo-code of HA

Proof Considering the equivalence between (2) and (3), we can see, for $i \in \mathcal{I}_k$ that

$$\left[V - \left(\min_{\alpha} B(\alpha)V - c_g(\alpha)\right)\right]_i = F_i(V).$$

Let us consider a vector $V \geq 0$ componentwise (here and in the following we will call 0 the null matrix). Using the monotony $H1$

$$(\mathbb{I} - B(\bar{\alpha}))V + c_g(\bar{\alpha}) \geq -\min_{\alpha\mathcal{A}} -c_g(\hat{\alpha})$$

where $\bar{\alpha} := \arg\min B(\alpha)V - c_g(\alpha)$. Now

$$B(\bar{\alpha})(V) \leq V, \quad \Rightarrow \quad B^{-1}(\bar{\alpha}) \geq \mathbb{I} \geq 0,$$

which is sufficient to prove the monotonicity of $B(\alpha)$ for the arbitrariness of the choice of V.

It is useful to underline the conceptual distinction between the convergence of the algorithm and the convergence of the numerical approximation to the continuous function v as discussed previously. In general, the Howard's algorithm is an acceleration technique for the calculus of the approximate solution, the error with the analytic solution will be depending from the discretization scheme used.

To conclude this introductory section let us make two monodimensional basic examples.

Example 1 (1D, Upwind scheme, Howard's Algorithm) An example for the matrix $B(\alpha)$ and the vector $c_g(\alpha)$ is the easy case of an upwind explicit Euler scheme in dimension one

$$\begin{cases} V_0 = g(x_0) \\ \lambda V_i = \min_{\alpha_i \in \mathcal{A}} \left(l(x_i, \alpha_i) + f_i^+(\alpha_i)\frac{V_{i+1}-V_i}{h} + f_i^-(\alpha_i)\frac{V_i - V_{i-1}}{h}\right), \quad i \in \{2, ..., N-1\} \\ V_N = g(x_N) \end{cases}$$

where x_i is a uniform discrete grid consisting in N knots of distance h. Moreover, $f_i^+(\alpha_i) = \max\{0, f(x_i, \alpha_i)\}$ and $f_i^-(\alpha_i) = \min\{0, f(x_i, \alpha_i)\}$. In this case the system (3) is

$$B(\alpha) = \begin{pmatrix} 1 + \frac{[f_1^+ - f_1^-]}{h\lambda} & -\frac{f_1^+}{h\lambda} & 0 & \cdots & 0 \\ \frac{f_2^-}{h\lambda} & 1 + \frac{[f_2^+ - f_2^-]}{h\lambda} & -\frac{f_2^+}{h\lambda} & \cdots & 0 \\ 0 & \ddots & \ddots & \ddots & 0 \\ 0 & \cdots & \cdots & \frac{f_N^-}{h\lambda} & 1 + \frac{[f_N^+ - f_N^-]}{h\lambda} \end{pmatrix}$$

and

$$c_g(\alpha) = \frac{1}{\lambda} \begin{pmatrix} -f_1^- \; g(x_0) + l(x_1, \alpha_1) \\ l(x_2, \alpha_2) \\ \vdots \\ l(x_{N-1}, \alpha_{N-1}) \\ +f_N^+ \; g(x_{N+1}) + l(x_N, \alpha_N) \end{pmatrix}$$

It is straightforward that the solution of Howard's algorithm, verifying $\min_\alpha B(\alpha)V - c_g = 0$, is the solution of (1).

Example 2 (1D, Semilagrangian, Howard's Algorithm) If we consider the standard 1D semiLagrangian scheme, the matrix $B(\alpha)$ and the vector $c_g(\alpha)$ are

$$B(\alpha) = \begin{pmatrix} 1 - e^{-\lambda h} b_1(\alpha_1) & -e^{-\lambda h} b_2(\alpha_1) & \cdots & -e^{-\lambda h} b_N(\alpha_1) \\ -e^{-\lambda h} b_1(\alpha_2) & 1 - e^{-\lambda h} b_2(\alpha_2) & \cdots & -e^{-\lambda h} b_N(\alpha_2) \\ \ddots & \ddots & \ddots & \ddots \\ -e^{-\lambda h} b_1(\alpha_N) & \cdots & -e^{-\lambda h} b_{N-1}(\alpha_N) & 1 - e^{-\lambda h} b_N(\alpha_N) \end{pmatrix}$$

and

$$c_g(\alpha) = \begin{pmatrix} hl(x_1, \alpha_1) + e^{-\lambda h} b_0(\alpha_1) g(x_0) \\ hl(x_2, \alpha_2) \\ \vdots \\ hl(x_{N-1}, \alpha_{N-1}) \\ hl(x_N, \alpha_N) + e^{-\lambda h} b_{N+1}(\alpha_N) g(x_{N+1}) \end{pmatrix}$$

and the coefficients b_i are the weights of a chosen interpolation $\mathbb{I}[V](x_i + hf(x_i, \alpha_j)) = \sum_{i=0}^{N+1} b_i(\alpha_j) V_i$.

Despite the good performances of the Policy Algorithm as a *speeding up* technique, in particular in presence of a convenient initialization (as shown for example in [14]) an awkward limit appears naturally: the necessity to store data of very big size.

Just to give an idea of the dimensions of the data managed it is sufficient consider that for a 3D problem solved on a squared grid of side n, for example, it would be essential to manage a $n^3 \times n^3$ matrix, task which becomes soon infeasible, with the increase of n. This give us an evident motivation to investigate the possibility to solve the problem in parallel, containing the complexity of the sub problems and the memory storage.

3 Domain Decomposition and Parallel version

The strict relation between the various points of the domain displayed by equation (14), makes the problem to find a parallel version of the technique, not an easy task to accomplish. The main problem, in particular, will be about passing information between the threads, necessary without a prior knowledge about the direction of the characteristics of the problem.

Our idea is to combine the policy iteration algorithm with a domain decomposition principle for HJ equations. Using the theoretical framework of the resolution of Partial Differential Equations on submanifolds, presented for example in [17,2], we consider a decomposition of Ω on a collection of subdomains:

$$\Omega := \bigcup_{i=1}^{M_\Omega} \Omega_i \bigcup_{j=1}^{M_\Gamma} \Gamma_j, \quad \text{with } \mathring{\Omega}_i \cap \mathring{\Omega}_j = \emptyset, \quad \text{for } i \neq j. \tag{4}$$

Where the interfaces Γ_j, $j = 1, \cdots, M_\Gamma$ are some strata of dimension lower than d defined as the intersection of two subdomains $\overline{\Omega}_i \cap \overline{\Omega}_k$ for $i \neq k$.

The notion of viscosity solution on the manifold, in this regular case, will be coherent with the definition elsewhere

Definition 1 A subsolution u on Γ is a usc function in Γ such that for any $\varphi \in C^1(\mathbb{R}^d)$ and any maximum point $x_0 \in \Gamma$ of $x \to u(x) - \varphi(x)$ is verified

$$\lambda\varphi(x_0) + H(x_0, D\varphi(x_0)) \leq 0,$$

the definition of supersolution is made accordingly.

Theorem 1 *Let us consider a domain decomposition as stated in* (4)*. The continuous function* $\overline{v} : \Omega \to \mathbb{R}$*, verifying in the viscosity sense of the system below*

$$\begin{cases} \lambda\overline{v}(x) + H(x, D\overline{v}(x)) = 0 & x \in \Omega_i, i = 1, ..., M_\Omega \\ \lambda\overline{v}(x) + H(x, D\overline{v}(x)) = 0 & x \in \Gamma_j, j = 1, ..., M_\Gamma, \\ \overline{v}(x) = g(x), & x \in \partial\Omega, \end{cases} \tag{5}$$

is coincident with the viscosity solution $v(x)$ *of* (14)*.*

Proof It is necessary to prove the uniqueness of a continuous viscosity solution for (5). After that, just invoking the existence and uniqueness results for the solution v (solution of the original problem), and observing that it is also a continuous viscosity solution of the system, from coincidence on the boundary, we get thesis.

To prove the uniqueness it is possible to use the classical argument of "doubling of variables". We recall the main steps of the technique for the convenience of the reader. For two continuous viscosity solutions $\bar{u}, \bar{v}$ of (5) using the auxiliary function

$$\Phi_\epsilon(x, y) := \bar{u}(x) - \bar{v}(y) - \frac{|x - y|^2}{2\epsilon}$$

which has a maximum point in (x_ϵ, y_ϵ), it is easy to see that

$$\max_{x\in\overline{\Omega}}(\bar{u}-\bar{v})(x) = \max_{x\in\overline{\Omega}}\Phi_\epsilon(x,x) \leq \max_{x,y\in\overline{\Omega}}\Phi_\epsilon(x,y) = \Phi_\epsilon(x_\epsilon, y_\epsilon)$$

now the limit

$$\liminf_{\epsilon\to 0^+}\Phi_\epsilon(x_\epsilon, y_\epsilon) \leq 0,$$

is proved as usual deriving Φ_ϵ and using the properties of sub supersolution, (for example, [1] Theo. II.3.1) with the observation that no additional problem appears when $(x_\epsilon, y_\epsilon) \in \Gamma_j$ because of the regularity of the Hamiltonian through Γ_j; for the possibility to exchange the role between $\bar{u}$ and $\bar{v}$ (both super and subsolutions) we have uniqueness.

In the following section we propose a parallel algorithm based on the numerical resolution of the decomposed system above. This technique consists of two steps iterations:

(i) Use Howard's algorithm to solve in parallel (n threads) the nonlinear systems obtained after discretization of (5) on the subdomains Ω_i (in this step the values of V are fixed on the boundaries);
(ii) Update the values of V on the interfaces of connection $\bigcup_j \Gamma_j$ by using Howard's algorithm on the nonlinear system obtained from the second equation of (5) (in this case the *interior points* of Ω_j are constant).

As it is shown later, this two-step iteration permits the transfer of information trough the interfaces performed by the phase (ii). This procedure, anyway, is not priceless, the number of the steps necessary for its resolution will be shown to be higher than the classic algorithm; the advantage will be in the resolution of smaller problems and the possibility of a resolution in parallel. Moreover, the coupling between phase (i) and (ii) produces a succession of results convergent in finite time, in the case of a finite space of controls.

The good performances of the algorithm, benefits and weak points will be discussed in details in Section 4.

3.1 Parallel Howard's Algorithm

To describe precisely the algorithm it is necessary state the following. Let us consider as before a uniform grid $G := \{x_j : j \in \mathcal{I}\}$, the indices set $\mathcal{I} := \{0, ..., N\}$, and a vector of all the controls on the knots $\alpha := (\alpha_1, ..., \alpha_N)^T \in \mathcal{A}^N$.

The domain Ω is decomposed as $\Omega := \cup_{i=1}^n \Omega_i \cup \Gamma$, where, coherently with above $\Gamma := \cup_{j=1}^{M_\Gamma} \Gamma_j$; this decomposition induces an similar structure in the indices set $\mathcal{I} := \mathcal{I}_1 \cup \mathcal{I}_2 \cup ... \cup ...\mathcal{I}_n \cup \mathcal{J}$, where every point x_k of index in $\mathcal{I}_i$ is an "*interior point*", in the sense that for every $x_j \in B_h(x_k)$ (ball centred in x_k of radius h, defined as previously), $j \notin \mathcal{I}_i$, for every $j \neq k$. The set $\mathcal{J}$ is the set of all the "*border points*", which means, for a $i \in \mathcal{J}$ we have that there exists at least two points x_j, $x_k \in B_h(x_i)$ such that $j \in \mathcal{I}_j$ and $k \in \mathcal{I}_k$ with $j \neq k$.

We will build n discrete subproblems on the subdomains Ω_i using as described before a monotone, stable and consistent scheme. In this case a discretization of the Hamiltonian provide, for every subdomain Ω_i, related to points x_j, $j \in \mathcal{I}_i$, a matrix $\hat{B}_i(\alpha_i)$ and a vector $\hat{c}_i(\alpha_i, \{V_j\}_{j\in\mathcal{J}})$, we highlighted here, the dependance of c_i from the border points which are, both, points where there are imposed the Dirichlet conditions (data of the problem) and points on the interface Γ which have to be estimed.

Assumed for simplicity that every $\mathcal{I}_i$ has the same number of k elements, called $\bar{k} := card(\mathcal{J})$, we have $k := \frac{N-\bar{k}}{n}$, and $\hat{B}_i(\cdot) \in \mathcal{M}_{k\times k}$, $\hat{c}_i(\cdot,\cdot) \in \mathbb{R}^k$.

In resolution over Γ we will have a matrix $\hat{B}_{n+1}(\alpha_{n+1})$ and a relative vector $\hat{c}_{n+1}(\alpha_{n+1}, \{V_j\}_{j\in\mathcal{I}\setminus\mathcal{J}})$, in the spaces, respectively, $\mathcal{M}_{\bar{k}\times\bar{k}}$ and $\mathbb{R}^{\bar{k}}$. (For the 1D case, e.g., we can easily verify that $\bar{k} = n-1$). In this framework, the numerical problem after the discretization of equations (5) is the following:

Find $V := (V_1, ..., V_i, ..., V_n, V_{n+1}) \in \mathbb{R}^N$ with $V_i = \{V_j \in \mathbb{R}^n \mid j \in \mathcal{I}_i\}$ for $i = 1, ..., n$ and $V_{n+1} = \{V_j \in \mathbb{R}^k \,|j \in \mathcal{J}\}$, solution of the following system of nonlinear equations:

$$\begin{cases} \min\limits_{\alpha_i\in\mathcal{A}^k} \Big(\hat{B}_i(\alpha_i)V_i - \hat{c}_i(\alpha_i, V_{n+1})\Big) = 0 & i = 1, ..., n \\ \min\limits_{\alpha_{n+1}\in\mathcal{A}^{\bar{k}}} \Big(\hat{B}_{n+1}(\alpha_{n+1})V_{n+1} - \hat{c}_{n+1}(\alpha_{n+1}, \{V_i\}_{i\in\{1,...n\}})\Big) = 0 \end{cases} \tag{6}$$

The resolution of first and the second equation of (6) will be called respectively *parallel part* and *iterative part* of the method. The resolution of the parallel and the iterative part will be performed alternatively, as a double step solver. So, the iteration of the algorithm will generate a sequence $V^s \in \mathbb{R}^N$ solution of the two steps system

$$\begin{cases} \min\limits_{\alpha\in\mathcal{A}^N} \big(B_i(\alpha)V^{s+2} - c_i(\alpha, V^{s+1})\big) = 0 \qquad i = 1, ..., n \\ \min\limits_{\alpha\in\mathcal{A}^N} \big(B_{n+1}(\alpha)V^{s+1} - c_{n+1}(\alpha, V^s)\big) = 0 \\ V^0 = V_0. \end{cases} \tag{7}$$

Where $B_i(\cdot)$, $c_i(\cdot,\cdot)$ are the matrices and vectors in $\mathcal{M}^{N\times N}$, and $\mathbb{R}^N$, containing $\hat{B}_i(\cdot)$, $\hat{c}_i(\cdot,\cdot)$ and such to return as solution the argument of $c_i(\alpha,\cdot)$ elsewhere. Evidently, $B_i(\cdot)$ $c_i(\cdot,\cdot)$ with $i \in \{1, ..., n\}$ are: equal to $\hat{B}_i$ in the $\{ik, .., (i+1)k - 1\} \times \{ik, .., (i+1)k - 1\}$ blocks, and equal to the raws $\mathcal{I}_i$ of the identity matrix elsewhere, $c_i = \hat{c}_i$ in the $\{ik, .., (i+1)k-1\}$ elements of the vector and $c_i(\cdot, V) = V$ elsewhere; the same, in the $\{nk+1, .., N\} \times \{nk+1, .., N\}$ block, $\{nk, .., N\}$ elements of the vector for $i = n+1$. It is clear that, despite this formal presentation, made to simplify the notation in the following, each equation of (7), negletting the trivial relations, is a nonlinear system on the same dimension than (6). Clearely, a solution of (6) is the fixed point of (7).

It is evident that such technique can be expressed as

$$\begin{cases} V_j^{s+2} = F_j^h(V^{s+2}, V^{s+1}) & j \in \mathcal{I}_i, \text{ with } i = 1, ..., n \\ V_j^{s+1} = F_j^h(V^{s+1}, V^s) & j \in \mathcal{J} \end{cases}$$

where, coherently with above $F_j^h(V,W) := \left[V + \min_{\alpha\in\mathcal{A}^N}(B_i(\alpha)V - c_i(\alpha,W))\right]_j$ for $j \in \mathcal{I}_i$.

From the assumptions on the discretization scheme some specific properties of $B_i(\cdot)$ and $c_i(\cdot,\cdot)$ can be derived

Proposition 2 *Let us assume $H1-H3$. Let state also*

(H4) *if $W_1 \geq W_2$ then $c_i(\alpha,W_1) \geq c_i(\alpha,W_2)$, for all $i = 1,...,n+1$, for all $\alpha \in \mathcal{A}$.*

Then there exists an unique solution of the system (7); moreover it holds true the following.

1. *The matrices $\hat{B}_i(\alpha)$ are monotone and not null for every $i \in \{1,...,n+1\}$, and for every $\alpha \in \mathcal{A}$.*
2. *If $\|V\|_\infty < +\infty$, we have that for all $i \in \{1,...,n+1\}$ and for every $\alpha \in \mathcal{A}$, there exists a $C > 0$ such that*
$$\|c_i(\alpha,V)\|_\infty \leq \frac{C}{\|B_i^{-1}(\alpha)\|_\infty}.$$
3. *Called V^* the fixed point of (7), if we have $V \leq V^*$ (resp. $V \geq V^*$) and $V = V^*$ in the idential arguments of c_i, then there exists a $\alpha \in \mathcal{A}$ such that*
$$B_i(\alpha)V - c_i(\alpha,V) \leq 0 \quad (\text{resp. } B_i(\alpha)V - c_i(\alpha,V) \geq 0) \quad \forall i = 1,...,n+1. \tag{8}$$

Proof To prove 1 the argument is the same of Proposition 1, just adding, (for $H4$) that $c_i(\cdot,V) \geq c_i(\cdot,0)$ because of the positivity of V. To prove 2, it is sufficient to note that for a $j \in \mathcal{I}_k$ the function F_j has the following form
$$F_j(V,W) = \left[B_k^{-1}(\bar{\alpha})c_k(\bar{\alpha},W)\right]_j$$
for a control $\bar{\alpha}$ minimizing a problem of the kind (3).

To prove 3, we notice that for a $V \leq V^*$,
$$0 = \min_\alpha B_i(\alpha)V^* - c_i(\alpha,V^*) = B_i(\bar{\alpha})V^* - c_i(\bar{\alpha},V)$$
where $\bar{\alpha} := \arg\min B_i(\alpha)V^* - c_i(\alpha,V^*)$; at the same time
$$0 = \min_\alpha B_i(\alpha)V^* - c_i(\alpha,V^*) \geq B_i(\bar{\alpha})V - c_i(\bar{\alpha},V^*)$$
then summing the two terms, the assumption is valid at least for $\bar{\alpha}$.

Here we introduce a convergence result for the (PHA) algorithm.

Theorem 2 *Assume that the function $\alpha \in \mathcal{A}^N \to B(\alpha) \in \mathcal{M}$ and $(\alpha,x) \in \mathcal{A}^N \times \mathbb{R}^n \to c(\alpha,x) \in \mathbb{R}^N$ are continuous on the variable α, $\mathcal{A}$ is a compact set of $\mathbb{R}^d$, and $(H1-H4)$ hold.*

Then there exists a unique V^* in $\mathbb{R}^N$ solution of (6). Moreover, the sequence V^k generated by the (PHA) (7) has the following properties:

Iterative Parallel Howard's Algorithm (PHA)

Inputs: $B_i(\cdot)$, $c_i\cdot, V^k_{n+1})$ for $i = 1, ..., n+1$
Initialize $V^0 \in \mathbb{R}^N$ and α^0.
Iterate $k \geq 0$:

1) *(Parallel Step)* for each $i = 1, ..., n$
   Call (HA) with inputs $B(\cdot) = B_i(\cdot)$ and $c_g(\cdot) = c_i(\cdot, \cdot)$
   Get $V^k_i = \{V^k(x_j)|j \in \mathcal{I}_i\}$.
2) *(Sequential Step)*
   Call (HA) with inputs $B(\cdot) = B_{n+1}(\cdot)$ and $c_g(\cdot) = c_{n+1}(\cdot, \{V^k_i\}i = \{1, ..., n\})$
   Get $V^k_{n+1} = \{V^k(x_j)|j \in \mathcal{J}\}$.
3) Compose the solution $V^{k+1} = (V^k_1, ..., V^k_n, V^k_{n+1})$
   If $\|V^{k+1} - V^k\|_\infty \leq \epsilon$ then *exit*, otherwise go to (1).

Outputs: V^{k+1}

Table 2 Pseudo-code of PHA

(i) Every element of the sequence V^s is bounded by a constant C, i.e. $\|V^s\|_\infty \leq C < +\infty$.
(ii) If $V^0 \leq V^*$ then $V^s \leq V^{s+1}$ for all $k \geq 0$.
(iii) $V^s \to V^*$ when s tends to $+\infty$.

Proof The existence of a solution comes directly from the monotonicity of the matrices $\hat{B}(\alpha)$, the existence of an inverse and then the existence of a solution of every system of 6. Let us show that such solution is limited as limit of a sequence of vectors of bounded norm. Observing that,

$$\|V^s\|_\infty = \max\left\{\|V^s_i\|_\infty\right\}_{i=1,...,n+1}$$

Without loss of generality we assume that $\|V^s\|_\infty \equiv \|V^s_{i^*}\|_\infty$. Calling

$$\hat{\alpha} := \arg\min_{\alpha \in \mathcal{A}} B_{i^*}(\alpha)V^s - c(\alpha, V^{s-1}),$$

the following bound is valid:

$$\|V^s\|_\infty = \|V^s_{i^*}\|_\infty \leq \|B^{-1}_{i^*}(\hat{\alpha})\|_\infty \|c_{i^*}(\hat{\alpha}, V^{s-1})\|_\infty.$$

Then, if V^{s-1} is bounded then $\|V^s\| \leq C$. Noting that $\|V^0\|$ is bounded, the thesis follows for induction.

Let us to pass now to prove the uniqueness. Let us take $V, W \in \mathbb{R}^N$ two solutions. Let us also define the vector W^* equal to V in the arguments non null of $c_i(\alpha, \cdot)$ and equal to W elsewhere, for a $i \in \{1, ..., n+1\}$. We have that, for a control β (for Proposition 2.3),

$$B_i(\beta)V - c_i(\beta, V) \geq 0 \geq B_i(\beta)W^* - c_i(\beta, W^*) = B_i(\beta)W - c_i(\beta, V)$$

then $\hat{B}_i(\beta)(V - W) \leq 0$ and for monotonicity $V \leq W$ for $j \neq i$. Exchanging the role of V and W, and for the arbitrariety of i we get the thesis.

(i) To prove that $V^k \in \mathbb{R}^N$ is an increasing sequence is sufficient to prove that taken $V_1, V_2 \in \mathbb{R}^N$ solution of

$$\min_{\alpha \in \mathcal{A}} B_i(\alpha)V_2 - c_i(\alpha, V_1) = 0$$

with (the opposite case is analogue) $V_2 \leq V^*$, for a choice of $i \in \{1, ..., n+1\}$ is such that $V_2 \geq V_1$. Let us observe, for a choice of $\beta \in \mathcal{A}$ as Prop. 2

$$\begin{aligned} 0 = \min_{\alpha \in \mathcal{A}} & B_i(\alpha)V_2 - c_i(\alpha, V_1) \\ & \leq B_i(\beta)V_2 - c_i(\beta, V_1) \leq B_i(\beta)V_2 - (B_i(\beta)V_1 - c_i(\beta, V_1)) - c(\beta, V_1) \end{aligned}$$

then $\hat{B}_i(\beta)(V_2 - V1) \geq 0$ then $V_2 \geq V_1$.

(ii) For monotony and boundedness of the sequence already shown we have the thesis.

It is also possible to show that the method stops to the fixed point in a finite time. As the Classic Howard's algorithm, this is an excellent feature of the technique; unfortunately, the estimate which is possible to guarantee is largely for excess and, although important from the theoretical point of view, not so effective to show the good qualities of the method. The performances will checked in the through some tests in the Section 4.

Proposition 3 *If $Card(A) < +\infty$ and convergence requests of Theorem 2 are verified, then (PHA) converges to the solution in less than $Card(A)^N$ iterative steps.*

Proof The proof is slightly similar to the classic Howard's case, [6].

Let us consider the abstract formulation $F : x \to y$, where $F(x)$ is determined by N_F parameter in A, and $G : y \to x$, where $G(y)$ is determined by N_G parameter in A. Then if we consider the iteration

$$\begin{aligned} F(x^k) &= y^k \\ G(y^k) &= x^{k+1} \end{aligned} \tag{9}$$

and we suppose (Theorem 2) $x^k \leq x^{k+1}$, $y^k \leq y^{k+1}$; than called α^k the $N_F + N_G$ variables in A associated to (x^k, y^k) we know that there exist a k and a l where $k < l \leq Card(A)^{N_F+N_G}$, such that $\alpha^k = \alpha^l$, and again $(x^k, y^k) = (x^l, y^l)$. Afterwards (x^k, y^k) is a fixed point of (9).
To restrict to our case is sufficient identify the process F with the (parallel) resolution on the sub-domains and G with the iteration on the surfaces.

Remark 1 It is worth to notice that the above estimation is worst than the Classical Howard's case. In fact, the classical algorithm find the solution in $Card(A)^N$, the (PHA) will have the same number of iterative steps. This number has to be multiplied, called M_1 the maximum number of nodes in a sub-domain and M_2 the number of nodes belonging to the interface, for $Card(A)^{(M_1+M_2)}$ getting, at the end, a total number of simple steps equal to $Card(A)^{(N+M_1+M_2)}$, much more than the classical case. In this analysis we do not consider anyway, the good point of decomposition techniques, the fact that any computational step is referred to a smaller and simpler problem, with the evident advantages in term of time elapsed in every threads and memory storage.

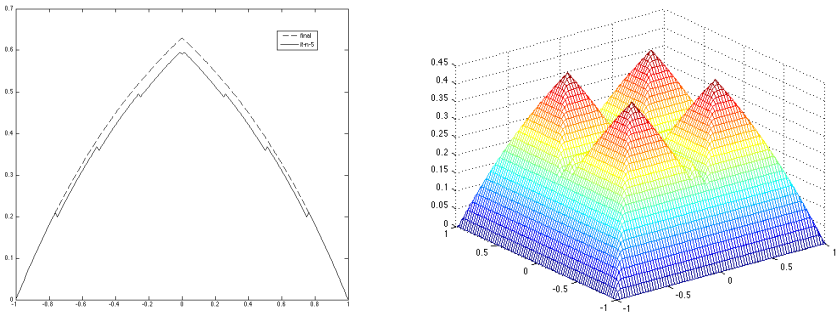

Fig. 1 Approximated solution of the iterative/parallel algorithm (left) in the 1D case, final time (dotted) and fifth iteration (solid), in the 2D case (right, 3rd iteration).

4 Performances, tuning parameters

The performances of the algorithm and its characteristics as speeding up technique will be tested in this section. Let us start with a standard academic example where, anyway, are present all the main characteristics of our technique.

1D problem. Consider the monodimensional problem

$$\begin{cases} u(x) + |Du(x)| = 1 \; x \in (-1, 1), \\ u(-1) = u(1) = 0. \end{cases} \tag{10}$$

It is well known that this equation (*Eikonal equation*) modelize the distance from the boundary of the domain, scaled by an exponential factor (*Kruzkov transform*, cf. [1]). Through a standard Euler discretization is obtained the problem in the form (3). In Table 4 is shown a comparison, in term of speed and efficacy, of our algorithm and the Classic Howard's one, in the case of a two thread resolution. It is possible appreciate as the parallel technique is not convenient in all the situations. This is due to the low number of parallel threads which are not sufficient to justify the construction. In the successive test, keeping fixed the parameter dx and tuning number of threads it is possible to notice how much influential is that variable in the speed of resolution.

Table 3 Testing performances, 1D. Our method compared with the classic Howard's with two sub-problems

2-splitting	Classic Howard's Alg.		Parallel Howard's Algorithm			
dx	time (s)	it.	t. (par. p.) (s)	it. (par.)	t. (it. p.)	Total t.
0.1	e-3	10	1e-4	4	1e-5	1e-3
0.05	6e-3	20	8e-4	5	e-5	3e-3
0.025	0.09	40	7e-3	6	2e-5	0.04
0.0125	0.32	80	0.048	8	1e-4	0.36
0.00625	2.22	160	0.34	14	8e-4	3.26

Table 4 Testing performances, 1D. Our method compared with the classic Howard's with various number of threads

dx=0.0125	Classic Howard's Alg.		Parallel Howard's Algorithm			
threads	t. (s)	it.	t. (par. p.) (s)	it. (par.)	t. (it. p.)	Total t.
2			0.48	4	1e-4	0.36
4			8e-3	6	1e-4	0.086
8	0.32	80	18e-4	7	6e-4	0.014
16			7e-4	10	4e-4	0.0095
32			2e-4	8	6e-3	0.011

In Table 4 we compare the iterations and the time necessary to reach the approximated solution, analysing the various phases of the algorithm, the time and the iterations necessary to solve every sub-problem (first two columns), the iterations and the time elapsed for the iterative part (which passes the information through the threads, next two columns), finally the total time. It is highlighted the optimal choice of number of threads (16 threard); it is evident as that number will change with the change of the discretization step dx. Therefore it is useful to remark that an additional work will be necessary to tune the number of threads accordingly to the needing of the problem; otherwise the risk is to is to loose completely the gain obtained through parallel calculus and to get worse performances even compared with the classical Howard's algorithm.

As in the rest of the paper all the codes are developed in Mathworks' MATLAB^{TM}and performed on a processor 2,8 Ghz Intel Core i7; in the tests the parallelization is simulated.

Table 5 Testing performances, 2D. Comparison with classical method and PH with 4 threads

4-threads	Cl. Howard's		Parallel Howard's Algorithm				
dx	time (s)	it.	t. (par. p.) (s)	it. (par.)	t. (it. p.)	it. (it. p.)	Total t.
0.1	0.05	11	0.009	8	0.02	2	0.04
0.05	2.41	21	0.05	13	0.03	2	0.14
0.025	73.3	40	2.5	22	0.15	3	7.83
0.0125	-	-	76	40	1.293	5	383.3

2D problem. The next test is in a space of higher dimension. Let us consider the approximation of the scaled distance function from the boundary of the square $\Omega := (-1,1) \times (-1,1)$, solution of the eikonal equation

$$\begin{cases} u(x) + \inf\limits_{a \in B(0,1)} \{-a \cdot Du(x)\} = 1 & x \in \Omega, \\ u(x) = 0 & x \in \partial\Omega. \end{cases} \tag{11}$$

where $B(0,1) \in \mathbb{R}^2$ is the usual unit ball. For the discretization of the problem is used a standard Euler discretization. Similar tests than the 1D case are performed, confirming the good features of our technique and, as already shown, the

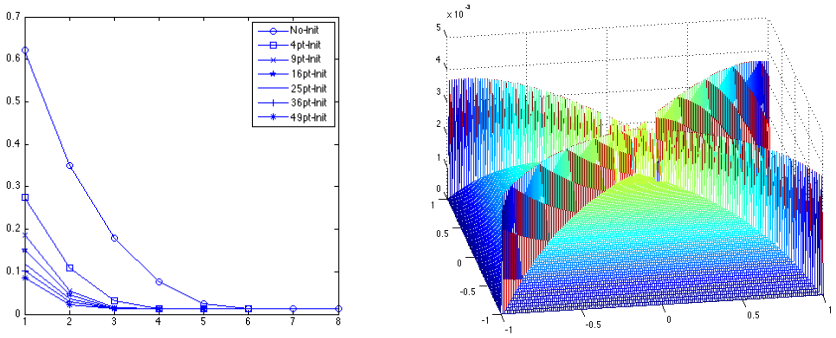

Fig. 2 Comparison with various initial guess to the speed of convergence of our method, in the L^2-norm (left) and distribution of the error $dx = 0.0125$, 16 threads (right).

necessity of an appropriate number of threads with respect to the complexity of the resolution.

In Table 5 performances of the Classic Howard's algorithm are compared with our technique. In this case the number of threads are fixed to 4; the Iterative-Parallel technique is evaluated in terms of: maximum time elapsed in one thread and max number of iterations necessary (first and second columns), time and number of iterations of the iterative part (third and fourth columns) and total time. In both the cases the control set $A := B(0,1)$ is substituted by a $32-$points discrete version. It is evident, in the comparison, an improvement of the speed of the algorithm even larger than the simpler 1D case. This justifies, more than the 1D case, our proposal.

Table 6 Testing performances, 2D. Comparing different choices of the number of threads

dx=0.025	Classic Howard's Alg.		Parallel Howard's Algorithm			
threads	t. (s)	it.	t. (par. p.) (s)	it. (par.)	t. (it. p.)	Total t.
4			2.5	22	0.15	7.83
9			0.9	18	0.5	5.08
16	73.3		0.05	13	1.6	$\boxed{1.826}$
25			0.03	12	2.4	2.52
36			0.016	11	6.04	6.11

In the Table 6 are compared the performances for various choices of the number of threads, for a fixed $dx = 0.025$. As in the 1D case is possible to see how an optimal choice of the number of threads can drastically strike down the time of convergence. In Figure 2 is possible to see the distribution of the error. As is predictable, the highest concentration will correspond to the non-smooth points of the solution. It is possible to notice also how our technique apparently does not introduce any additional error in correspondence of the interfaces connecting the sub-domains.

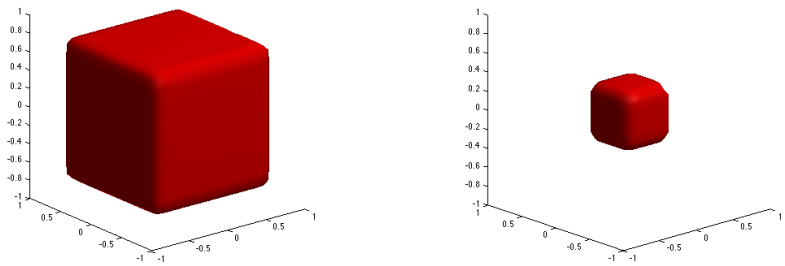

Fig. 3 Two level sets (corresponding to levels $u(x) = 0.192$ (left) $u(x) = 0.384$ (right) of the approximated solution obtained with a $dx = 0.1$ and an 8−threads PHA.

Remark 2 As shown in the tests, an important point of weakness of our technique is represented by the iterative part, which is smaller and therefore easier than the ones solved in the parallel part, but it is highly influential in terms of general performances of the algorithm. In particular the number of the iterations of the coupling iterative-parallel part is sensible to a good initialization of the "internal boundary" points. As is shown in Figure 2 a right initialization, even obtained on a very coarse grid, is highly influential in the number of performances. In this section, all the tests are made with a initialization of the solution on a 16-point grid.

Table 7 Testing performances, 3D. Comparison with classical method and PI-H with 8 threads

8-threads	Cl. Howard's		Iterative-Parallel Howard's Algorithm				
dx	time (s)	it.	t. (par. p.) (s)	it. (par.)	t. (it. p.)	it. (it. p.)	Total t.
0.4	0.004	4	0.003	4	0.002	1	0.05
0.2	0.22	6	0.026	6	0.016	2	0.052
0.1	164.2	11	1.102	8	2.1	4	6.78
0.05	-	-	164	10	4.98	3	494

3D problem. Analogue results are obtained also in a 3D resolution. Of course the effects of the increasing number of control points will limit the possibility of a fine discretization of the domain.
Let us consider the domain $\Omega := [-1, 1]^3$ and the equation (11), where $A := B(0, 1)$, unitary ball in R^3. In Figure 3 there are shown two level sets of the solution obtained. A comparison with the performances of the Classic Howard's algorithm are shown in Table 6.

Remark 3 With the growth of the dimensionality of the problem a special care should be dedicated to the resolution of the iterative step. Suppose to simplify the

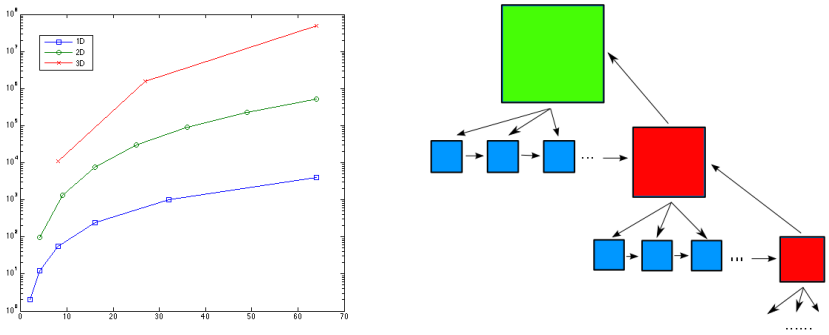

Fig. 4 Optimal number of splitting for number of variables in the discretization (left) and iterative structure of the algorithm (right) to reduce the original problem (green) to a fixed number of variables sub-problems (blue).

procedure considering a square domain (in dimension $d = 1, 2, 3, ..$ an interval, a square, a cube..) and a successive splitting in square subdomains. Calling N the number of total variables and N_s the number of the splitting (which generates a division in N_s^d subdomains) the number of the elements in every threads of the parallel part is $\frac{N}{(N_s)^d}$, and the number of the variables in the iterative part $\frac{N}{\sqrt[d]{N}}(N_s - 1)d$. Clearly the optimal choice of the number of threads is such that the elements of the iterative part are in the same number of each subdomain, so it is straight forward to find the following optimal relation between number of splitting and total elements

$$N = \left(N_s^d (N_s - 1)d\right)^d$$

It is evident that for a high number of elements, (Figure 4) the number of threads to be optimal is seriously limited, in particular in the high dimensional case, provoking an high number of elements in every threads. This reduces noteworthy the efficacy of the parallelization. The problem can be overcome with an additional parallel decomposition (sketched in Figure 4) of the iterative pass, permitting us to reduce each subproblem to a complexity acceptable.

5 Extensions and Special Cases

In this section are shown some non trivial extensions to more general situations of the method. We will show, in particular, how to adapt the parallelization procedure to the case of a target problem, an obstacle problem and max-min problems, where, we recall the special structure of the Hamiltonian requires some cautions and remarks.

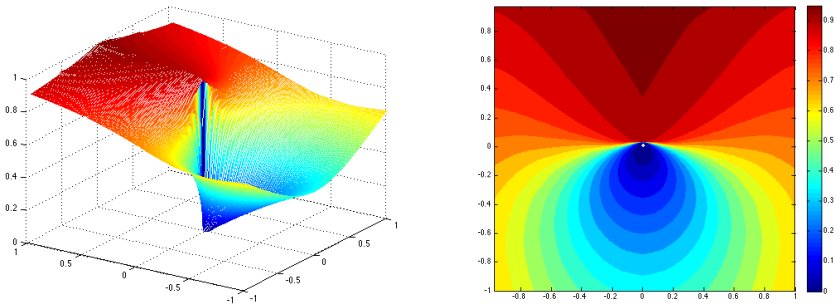

Fig. 5 Approximated solution for the Zermelo's navigation problem $dx = 0,01$.

5.1 Target problems

An important class of problems where is useful to extend the techniques discussed is the Target problems, complementary problem of the discussed exit-problems, where a trajectory is driven to arrive in a *Target set* $\mathcal{T} \subset \Omega$ optimizing a cost functional.

A easy way to modify our Algorithm to this case is to change the construction procedure for B and C:

$$\left[B'(\alpha)\right]_i := \begin{cases} [B(\alpha)]_i\,, & \text{if } x_i \notin \mathcal{T} \\ [\mathbb{I}]_i\,, & \text{otherwise;} \end{cases} \qquad c'(\alpha)_i := \begin{cases} c(\alpha)_i, & \text{if } x_i \notin \mathcal{T} \\ 0, & \text{otherwise;} \end{cases} \tag{12}$$

this, with the classical further construction of *ghost nodes* outside the domain Ω to avoid the exit of the trajectories from Ω, will solve this case.

Example 3 (Zermelo's Navigation Problem) Another well known benchmark in the field is the so-called Zermelo's navigation problem, the main problem, in this case, is that the dynamic is driven by a force of comparable power with respect to our control. The target to reach will be a ball of radius equal to 0.005 centred in the origin

$$f(x,a) = a + \begin{pmatrix} 1 - x_2^2 \\ 0 \end{pmatrix}, \quad \Omega = [-1,1]^2, \quad A = B(0,1), \quad \lambda = 1, \quad l(x,y,a) = 1. \tag{13}$$

In Table 8 a comparison with the number of threads chosen is made. Now we are in presence of characteristics not aligned with the grid, but the performances of the method are poorly effected. Convergence is archived with performances absolutely comparable with the ones already described for the Eikonal Equation.

Table 8 Zermelo's navigation problem. Comparison of various choices of the number of threads

dx=0.025	Classic Howard's Alg.		Parallel Howard's Algorithm			
threads	t. (s)	it.	t. (par. p.) (s)	it. (par.)	t. (it. p.)	Total t.
4			1.31	11	0.13(4)	5.4
9			0.7	9	0.7(5)	4.2
16	37.9	20	0.031	7	1.38(5)	1.53
25			0.02	7	2.7(6)	3.9
36			0.01	8	5.19(7)	5.28

5.2 Obstacle Problem

Dealing with a optimal problem with constraints various approaches have been proposed. In this section we will consider an implicit representation of the constraints through a level-set function. Let us to consider the general single obstacle problem

$$\begin{cases} \max\left(\lambda v(x) + H(x, Dv(x)), v(x) - w(x)\right) = 0 & x \in \Omega \\ v(x) = g(x) & x \in \partial\Omega \end{cases} \tag{14}$$

where the Hamiltonian H is of the form discussed in Section 2 and the standard hypothesis about regularity of the terms involved are verified. The distinctive trait of this formulation is about the term $w(x) : \Omega \to \mathbb{R}$, assumed regular, typically stated as the opposite of the signed distance from the boudary of a subset $K \subset \Omega$. The solution of this problem is coincident, where defined, with the solution of the same problem in the space in the space $\Omega \setminus K$, explaining the name of "obstacle problem" (cf. [8]).

Through an approximation of the problem in a finite dimensional one, in a similar way as already explained, is found the following variation of the Howard's problem

$$\text{Find } V \in \mathbb{R}^N; \quad \min_{\alpha \in \mathcal{A}^N} \min(B(\alpha)V - c(\alpha), V - W) = 0, \tag{15}$$

where the term W is a sampling of the function w on the knot of the discretization grid.

It is direct to show that changing the definition of the matrix B and c, is possible to come back to the problem (3). Adding an auxiliary control to the set $\mathcal{A}' := \mathcal{A} \times \{0, 1\}$ and re-defying the matrices B and c as

$$\left[B'(\alpha)\right]_i := \begin{cases} \left[B(\alpha)\right]_i, & \text{if } B(\alpha)V - c(\alpha) \geq V - W \\ \left[\mathbb{I}\right]_i, & \text{otherwise;} \end{cases} \tag{16}$$

$$c'(\alpha)_i := \begin{cases} c(\alpha)_i, & \text{if } B(\alpha)V - c(\alpha) \geq V - W \\ W_i, & \text{otherwise;} \end{cases} \tag{17}$$

$$\text{for } i = 1, ..., N$$

(where the X_i is the $i-$row if X is a matrix, and the $i-$ element if X is a vector, and $\mathbb{I}$ is the identity matrix), the problem becomes

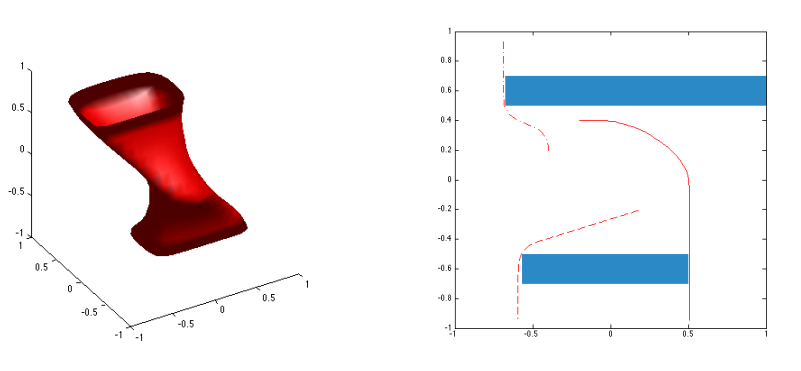

Fig. 6 Value function of Dubin Car Problem (left, free of constraints) and some optimal trajectories in the case with constraints (right, the *Target* is $y = \{-1, 1\}$).

$$\text{Find } V \in \mathbb{R}^N; \quad \min_{\alpha \in \mathcal{A}'} (B'(\alpha)V - c'(\alpha)) = 0 \tag{18}$$

which is in the form (3). To ensure the convergence of the (PHA) Algorithm is necessary to verify the Hypothesis of Theorem 2.

Remark 4 The verification of conditions of convergence in the obstacle problem are often trivially derived from the free of constraints case. For example if we have that the matrix $B(\alpha)$ is strictly dominant (i.e. $A_{ij} \leq 0$ for every $j \neq i$, and there exists a $\delta > 0$ such that for every i, $A_{ii} \geq \delta + \sum_{i \neq j} |A_{ij}|$), then the properties of the terms are automatically verified, (i.e. since all $B_i(\alpha)$ are strictly dominant and thus monotone).

Example 4 (Dubin Car with obstacles)

A classical problem of interest is the optimization of trajectories modelled by

$$f(x, y, z, a) := \begin{pmatrix} c\cos(\pi z) \\ c\sin(\pi z) \\ a \end{pmatrix}$$

which produce a collection of curves in the plane (x, y) with a constraint in the curvature of the path. Typically this is a simplified model of a car of constant velocity c with a control in the steering wheel.
The value function of the exit problem from the domain $\Omega := (-1, 1)^2$, $A = [-1, 1]$ discretized uniformly in 8 points is presented in Figure 6. It is straight forward to imagine the same problem with the presence of constraints. That problem can be handled with the technique described above producing the results shown in the same Figure 6.

PHA (MaxMin case)

Initialize $V^0 \in \mathbb{R}^N$ α^0 for all $i \in \{1, ..., n+1\}$.
k:=1;

1) Iterate *(Parallel Step)* for every $i = 1, ..., n$ do:
   $s := 0$
   1.i) Find $V_i^s \in \mathbb{R}^n$ solution of $F_i^\beta(V_i^s) = 0$.
   If $s \geq 1$ and $V_i^s = V_i^{s-1}$, then $V_i := V_i^s$, and exit (from inner loop).
   Otherwise go to (1.ii).
   1.ii) $\beta_i^{s+1} := \arg\min_{\alpha \in \mathcal{A}^n} F_i^\beta(V_i^s) = 0$.
   Set $s := s+1$ and go to (1.i)
2) Iterate *(Sequential Step)* for $t \geq 0$
   2i) Find $V_{n+1}^t \in \mathbb{R}^h$ solution of $F_{n+1}^\beta(V_{n+1}^t) = 0$.
   If $t \geq 1$ and $V_{n+1}^t = V_{n+1}^{t-1}$, then $V_{n+1} = V_{n+1}^t$, and go to (3).
   Otherwise go to (2ii).
   2ii) $\beta_{n+1}^{t+1} := \arg\min_{\beta_{n+1} \in \mathcal{B}^h} F_{n+1}^\beta(V_{n+1}) = 0$.
   Set $t := t+1$ and go to (2i)
3) Compose the solution $V^{k+1} = (V_1, V_2, ..., V_n, V_{n+1})$
   k:=k+1;
   If $V^{k+1} = V^k$ then *exit*, otherwise go to (1).

Table 9 Pseudo-code of PHA for MaxMin problems.

5.3 Max-min Problems

Another natural extension of the Howard's problem (3) is about max-min problems of the form

$$\text{Find } V \in \mathbb{R}^N; \quad \max_{\beta \in \mathcal{B}^N} \left(\min_{\alpha \in \mathcal{A}^N} (B(\alpha, \beta)V - c(\alpha, \beta)) \right) = 0. \tag{19}$$

Such a non linear equations arises in differential games setting and in robust control, for example. Also in this case, a modified version of the policy iteration algorithm can be shown to be convergent (cf. [6]). Our aim in this subsection is to show how even a parallel version of the procedure is possible.

Let $\mathcal{A}$ and $\mathcal{B}$ be two compact sets, and let us consider the differential problem (14) with the Hamiltonian $H(x,p) := \inf_{\beta \in \mathcal{B}} \sup_{\alpha \in \mathcal{A}} \{-f(x,a,b) \cdot p - l(x,a,b)\}$. There exists a wide literature about the study of this problem, but for our purposes, it is sufficient consider the case where the viscosity solutions of this problem is unique, and the *Isaacs' condition* ($H(x,p) = \overline{H}(x,p) := \sup_{\alpha \in \mathcal{A}} \inf_{\beta \in \mathcal{B}} \{-f(x,a,b) \cdot p - l(x,a,b)\}$) is verified. For a whole presentation of the subject we recall the monographs [3],[1].

As in section 3 we introduce a decomposition of the domain Ω and using again Theorem 1, which needs just the existence and uniqueness of the solution, we can state an analogue result to (5). This give us the authorization to proceed with a decomposed version of the problem(19).

We also introduce the function $F_i^\beta : \mathbb{R}^n \to \mathbb{R}$, for $\beta \in \mathcal{B}^n$ and $i \in \mathcal{I}$ defined by

$$F_i^\beta(V) := \min_{\alpha \in \mathcal{A}^n} (B(\alpha, \beta)V - c(\alpha, \beta, V) \tag{20}$$

It is evident that the problem (19), in analogy with the previous case, is equivalent to solve the following system of nonlinear equations

$$\begin{cases} \min\limits_{\beta \in \mathcal{B}^k} F_i^\beta(V_i) = 0 & i = 1, ..., n \\ \min\limits_{\beta \in \mathcal{B}^h} F_{n+1}^\beta(V_{n+1}) = 0 \end{cases} \tag{21}$$

The Iterative-Parallel Version of the Howard Algorithm in the case of a maxmin problem is summarized in Table 9.

Remark 5 It is worth to notice that at every call of the function F^β is necessary to solve a minimization problem over the set $\mathcal{A}$, this can be performed in an approximated way, using, for instance, the classical Howard's algorithm. This gives to the dimension of this set a big relevance on the performances of our algorithm. For this reason, if the cardinality of $\mathcal{A}$ (in the case of finite sets) is bigger than $\mathcal{B}$, it is worth to pass to the alternative problem $-max_{\alpha \in \mathcal{A}} \min_{\beta \in \mathcal{B}}(B(\alpha, \beta)V - c(\alpha, \beta))$ (here are used the Isaacs' conditions) before the resolution, inverting in this way, the role of $\mathcal{A}$ and $\mathcal{B}$ in the resolution.

Example 5 (A Pursuit-Evasion game) One of the most known example of max-min problem is the Pursuit evasion game; where two agents have the opposite goal to reduce/postpone the time of capture. One of the most simple contest of that problem is related to a dynamic

$$f(x, y, z, a, b) := \begin{pmatrix} a_1/2 - b_1 \\ a_2/2 - b_2 \end{pmatrix}$$

controls are taken in the unit ball $A = B = B(0, 1)$ and capture happens when the trajectory is driven to touch the small ball $B(0, \rho)$, ($\rho = 0.15$, in this case). The passage to a Target problem is managed as described previously. In Figure 7 the approximated value function of that problem is shown.

6 Conclusions

References

1. M. Bardi and I. Capuzzo-Dolcetta, *Optimal Control and Viscosity Solution of Hamilton-Jacobi-Bellman Equations*. Birkhauser, Boston Heidelberg, 1997.
2. G. Barles, A. Briani and E. Chasseigne, A Bellman approach for two-domains optimal control problems in $\mathbb{R}^N$. ESAIM Contr. Op. Ca. Va., Vol. 19 n. 03 (2013), pp. 710–739.
3. M. Bardi, T.E.S. Raghavan, T. Parthasarathy, *Stochastic and Differential Games: Theory and Numerical Methods*, Birkhäuser, Boston, 1999.
4. R.C. Barnard and P.C. Wolenski, Flow Invariance on Stratified Domains, Set-Valued Var. Anal., 21 (2013) pp. 377–403.

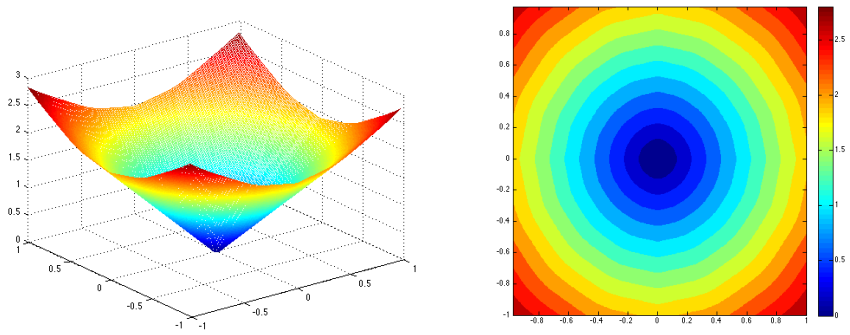

Fig. 7 Approximated solution of the Pursuit Evasion game, $dx = 0.0125$

5. R. Bellman, *Dynamic Programming*, Princeton University Press, Princeton, NJ, 1957.
6. O. Bokanowski, S. Maroso and H. Zidani, Some convergence results for Howard's algorithm, SIAM J. Numer. Anal. Vol 47, n. 4, (2009), pp. 3001–3026.
7. F. Camilli, M. Falcone, P. Lanucara and A. Seghini, A domain decomposition Method for Bellman Equations, Cont. Math. 180, (1994) pp. 477–483.
8. F. Camilli, P. Loreti and N. Yamada, Systems of convex Hamilton-Jacobi equations with implicit obstacles and the obstacle problem, Comm. Pure App. Math., vol. 8, no.(2009), p. 1291–1302.
9. Y. Cheng, Yingda and C.-W. Shu. A discontinuous Galerkin finite element method for directly solving the Hamilton-Jacobi equations. Journal of Computational Physics 223 1 (2007): 398–415.
10. M. G. Crandall and P. L. Lions Two Approximations of Solutions of Hamilton-Jacobi Equations, Math. Comp. Vol. 43 n. 167 (1984) pp. 1–19.
11. L.C. Evans, *Partial differential equations: Graduate studies in Mathematics.* American Mathematical Society 2, 1998.
12. M. Falcone and R. Ferretti, *Semi-Lagrangian Approximation Schemes for Linear and Hamilton-Jacobi Equations*, Applied Mathematics series, SIAM, 2013.
13. R.A. Howard, *Dynamic Programming and Markov Processes*, The MIT Press, Cambridge, MA, 1960.
14. A. Alla, M. Falcone, and D. Kalise. An efficient policy iteration algorithm for dynamic programming equations. PAMM 13.1 (2013): 467–468.
15. L. Qi, *Convergence analysis of some algorithms for solving nonsmooth equations*, Math. Oper. Res. 18 (1993): 227–244.
16. L. Qi and J. Sun, *A nonsmooth version of Newton's method*, Math. Program., 58 (1993): 353–367.
17. Z. Rao and H. Zidani, Hamilton-Jacobi-Bellman Equations on Multi-Domains, Birkhauser Basel. Volume 164, (2013).
18. sc M. Santos and J. Rust. Convergence properties of policy iteration. SIAM Journal on Control and Optimization 42 6 (2004): 2094-2115.
19. P. Souganidis, Approximation schemes for viscosity solutions of Hamilton-Jacobi equations. Journal of differential equations 59 1 (1985): 1–43.
20. M. Sun, Domain Decomposition algorithms for solving Hamilton Jacobi-Bellman equations, Num. Funct. Analysis Opt. 14, (1993) pp.145–166.
21. M. Puterman and S. L. Brumelle. On the convergence of policy iteration in stationary dynamic programming." Mathematics of Operations Research 4.1 (1979): 60–69.
22. S.Z. Zhou and W.P. Zhan, A new domain decomposition method for an HJB equation. J. Comput. Appl. Math. 159(1), (2003) pp. 195–204.