A Dijkstra-type algorithm for dynamic games

Martino Bardi [1] Juan Pablo Maldonado Lopez [2]

[1]Dipartimento di Matematica, Università di Padova, Italy

[2]formerly at Combinatoire et Optimisation, UPMC, France

Numerical methods for PDEs: optimal control, games and image processing
Roma, December 4-5, 2014

Dedicated to Maurizio Falcone on his 60th birthday

Who is Dijkstra and what do we want from him?

- The Dijkstra algorithm (1959) is a classical tool for finding shortest paths on finite graphs,
- it has low computational complexity: running time $= O(e + v \log v)$, v=number of vertices , e=number of edges

Who is Dijkstra and what do we want from him?

- The Dijkstra algorithm (1959) is a classical tool for finding shortest paths on finite graphs,
- it has low computational complexity: running time $= O(e + v \log v)$, v=number of vertices , e=number of edges
- reasons: it is "single pass", i.e., once a node is accepted the value is not recomputed on it, and terminates in a finite number of steps,
- it is the basis for Fast Marching Methods in optimal control (Tsitsiklis 95) and in front propagation (Sethian 96, book 99).

Who is Dijkstra and what do we want from him?

- The Dijkstra algorithm (1959) is a classical tool for finding shortest paths on finite graphs,
- it has low computational complexity: running time $= O(e + v \log v)$, v=number of vertices , e=number of edges
- reasons: it is "single pass", i.e., once a node is accepted the value is not recomputed on it, and terminates in a finite number of steps,
- it is the basis for Fast Marching Methods in optimal control (Tsitsiklis 95) and in front propagation (Sethian 96, book 99).
- Question 1: can it be adapted to discrete games?

Who is Dijkstra and what do we want from him?

- The Dijkstra algorithm (1959) is a classical tool for finding shortest paths on finite graphs,
- it has low computational complexity: running time $= O(e + v \log v)$, v=number of vertices , e=number of edges
- reasons: it is "single pass", i.e., once a node is accepted the value is not recomputed on it, and terminates in a finite number of steps,
- it is the basis for Fast Marching Methods in optimal control (Tsitsiklis 95) and in front propagation (Sethian 96, book 99).
- Question 1: can it be adapted to discrete games?
- Question 2: can it be used for the numerical solution of differential games and of Hamilton-Jacobi equations with non-convex Hamiltonian?

Related refs.: Q1: Alfaro, Henzinger, Kupferman 07;
Q2: vonLossow 07, Grüne-Junge 08, Cristiani 09, Cacace-Cristiani-Falcone 12.

Zero-sum discrete dynamic games

- Two players choose simultaneously and independently actions at each instant of time. They know the costs they incurr, they remember the (perfectly observed) past, and they are aware of each other's goals.
- Shapley (1953) proved that the value of finite state, discrete time, discounted stochastic games satisfies a dynamic programming principle and is the unique fixed point of a contractive operator.
- Many methods to compute the value are more or less variants of more general algorithms to compute fixed points, see the survey by Filar and Raghavan (ZOR 1991), or Kushner (IEEE Trans. Autom. Control 2004).
- A Dijkstra-type algorithm makes sense for positive running costs. We will also assume alternating moves, instead of simultaneous moves.

The model

- Let $\mathcal{X}$ be a finite set (the state space), A, B be finite action sets for player 1 and 2 respectively.
- For a deterministic transition function $S : \mathcal{X} \times A \times B \to \mathcal{X}$ define the trajectory $x_\bullet = x_\bullet(x, a_\bullet, b_\bullet)$ recursively by

$$x_{n+1} = S(x_n, a_n, b_n), \quad x_0 = x.$$

- Let $\mathcal{X}_f \subset \mathcal{X}$, denote a *terminal set* of nodes (which player 1 wishes to attain) and let $\gamma \in (0, 1]$ be a discount factor.
- The *arrival time* $\hat{n} : \mathcal{X} \times A^{\mathbb{N}} \times B^{\mathbb{N}} \to \mathbb{R}$ is

$$\hat{n}(x, a_\bullet, b_\bullet) = \begin{cases} \min\{n \in \mathbb{N} : x_n \in \mathcal{X}_f\}, & \text{if } \{n \in \mathbb{N} : x_n \in \mathcal{X}_f\} \neq \emptyset \\ +\infty & \text{else,} \end{cases}$$

- The *running* and *terminal* cost (for player 1)

$$\ell : \mathcal{X} \times A \times B \to \mathbb{R}, \quad 0 < \ell_0 \leq \ell(x,a,b) \leq L,$$
$$g : \mathcal{X}_f \to \mathbb{R}, \quad g_0 \leq g(x) \leq g_1, \forall x \in \mathcal{X}_f$$

- Total cost: $J : \mathcal{X} \times A^{\mathbb{N}} \times B^{\mathbb{N}} \to \mathbb{R}$, with $0 < \gamma \leq 1$,

$$J(x, a_\bullet, b_\bullet) := \sum_{n=0}^{\hat{n}-1} \ell(x_n, a_n, b_n)\gamma^n + \gamma^{\hat{n}} g(x_{\hat{n}}).$$

- Example: for $\ell \equiv 1$, $g \equiv 0$, $\gamma = 1$,

  $J =$ number of steps to reach the target.

The game: alternating moves

We consider the case when player 1 chooses his action after player 2.

Definition

A map $\alpha : B^{\mathbb{N}} \to A^{\mathbb{N}}$ *is a* non-anticipating strategy *for player 1 if*

$$b_n = \tilde{b}_n, \forall n \leq m \implies \alpha[b_\bullet]_n = \alpha[\tilde{b}_\bullet]_n, \forall n \leq m,$$

and we denote $\alpha \in \mathcal{A}$.

This allows us to introduce the *lower value function*

$$V^-(x) := \inf_{\alpha \in \mathcal{A}} \sup_{b_\bullet \in B^{\mathbb{N}}} J(x, \alpha[b_\bullet], b_\bullet).$$

The game: alternating moves

We consider the case when player 1 chooses his action after player 2.

Definition

A map $\alpha : B^{\mathbb{N}} \to A^{\mathbb{N}}$ *is a* non-anticipating strategy *for player 1 if*

$$b_n = \tilde{b}_n, \forall n \leq m \implies \alpha[b_\bullet]_n = \alpha[\tilde{b}_\bullet]_n, \forall n \leq m,$$

and we denote $\alpha \in \mathcal{A}$.

This allows us to introduce the *lower value function*

$$V^-(x) := \inf_{\alpha \in \mathcal{A}} \sup_{b_\bullet \in B^{\mathbb{N}}} J(x, \alpha[b_\bullet], b_\bullet).$$

The upper value function can be defined in a completely analogous way and corresponds to the game where player 2 knows in advance the move of player 1.

Dynamic programming

Proposition

The lower value function satisfies

$$\begin{aligned} V^-(x) &= g(x),\ \forall x \in \mathcal{X}_f, \\ V^-(x) &= \max_{b\in B}\min_{a\in A}\left\{\ell(x,a,b) + \gamma V^-(S(x,a,b))\right\},\ \forall x \notin \mathcal{X}_f, \\ V^-(x) &= \inf_{\alpha\in\mathcal{A}}\sup_{b_\bullet\in B^{\mathbb{N}}}\left\{\sum_{n=0}^{k\wedge\hat{n}-1}\ell(x_n,\alpha[b_\bullet]_n,b_n)\gamma^n + \gamma^{k\wedge\hat{n}}V^-(x_{k\wedge\hat{n}})\right\},\ \forall k. \end{aligned}$$

Proposition (Dynamic programming "for sets")

(DPS) Let $\mathcal{X}_f \subset \tilde{\mathcal{X}} \subset \mathcal{X}$ *and let* $\tilde{n}$ *denote the arrival time to* $\tilde{\mathcal{X}}$*, i.e.* $\tilde{n} = \tilde{n}(x, a_\bullet, b_\bullet) = \inf\{n \in \mathbb{N} : x_n \in \tilde{\mathcal{X}}\}$. *Then*

$$V^-(x) = \inf_{\alpha\in\mathcal{A}}\sup_{b_\bullet\in B^{\mathbb{N}}}\left\{\sum_{n=0}^{\tilde{n}-1}\ell(x_n,\alpha[b_\bullet]_n,b_n)\gamma^n + \gamma^{\tilde{n}}V^-(x_{\tilde{n}})\right\}.$$

The algorithm

Require: $n = 0, \mathrm{Acc}_0 := \mathcal{X}_f,\ W_0(x) := +\infty, \forall x \in \mathcal{X},$
$V_0^-(x) = g(x), \forall x \in \mathcal{X}_f$

while $\mathrm{Acc}_n \neq \mathcal{X}$ **do**

  for $x \in \mathcal{X} \setminus \mathrm{Acc}_n, b \in B$ **do**

    $A_n(x, b) := \{a \in A : S(x, a, b) \in \mathrm{Acc}_n\}$

  end for

end while

The algorithm

Require: $n = 0, \mathrm{Acc}_0 := \mathcal{X}_f,\ W_0(x) := +\infty, \forall x \in \mathcal{X},$
$V_0^-(x) = g(x), \forall x \in \mathcal{X}_f$
while $\mathrm{Acc}_n \neq \mathcal{X}$ **do**
  for $x \in \mathcal{X} \setminus \mathrm{Acc}_n, b \in B$ **do**
    $A_n(x,b) := \{a \in A : S(x,a,b) \in \mathrm{Acc}_n\}$
  end for
end while
$\mathrm{Cons}_n := \{x \in \mathcal{X} \setminus \mathrm{Acc}_n : A_n(x,b) \neq \emptyset\ \forall b \in B\}$
while $\mathrm{Cons}_n \neq \emptyset$ **do**
  $W_{n+1}(x) :=$
$\max_{b\in B} \min_{a \in A_n(x,b)} \{\ell(x,a,b) + \gamma V_n^-(S(x,a,b))\},\ \forall x \in \mathrm{Cons}_n$
  $\mathrm{Acc}_{n+1} := \mathrm{Acc}_n \cup \mathrm{argmin}\, W_{n+1}$
  $V_{n+1}^-(x) := W_{n+1}(x),\ \forall x \in \mathrm{argmin}\, W_{n+1}$
  $V_{n+1}^-(x) := V_n^-(x), \forall x \in \mathrm{Acc}_n$
  $n \leftarrow n+1$
end while

Note that Acc_n is strictly increasing as long as $\mathrm{Cons}_n \neq \emptyset$, so the algorithm terminates in a finite number N of steps, at most $|\mathcal{X} \setminus \mathcal{X}_f|$ = the cardinality of $\mathcal{X} \setminus \mathcal{X}_f$.

Note that Acc_n is strictly increasing as long as $\mathrm{Cons}_n \neq \emptyset$, so the algorithm terminates in a finite number N of steps, at most $|\mathcal{X} \setminus \mathcal{X}_f| =$ the cardinality of $\mathcal{X} \setminus \mathcal{X}_f$.

For the convergence we consider the set $\mathcal{R}$ of nodes from which player 1 can reach the terminal set for any behavior of player 2, i.e.,

$$\mathcal{R} := \{x \in \mathcal{X} : \inf_{\alpha \in \mathcal{A}} \sup_{b_\bullet \in B^{\mathbb{N}}} \hat{n}(x, \alpha[b_\bullet], b_\bullet) < +\infty\},$$

called the reachability set (by player 1).

Convergence of the algorithm

Condition

(Condition C) If $\gamma < 1$

$$L + \gamma g_1 \leq \frac{\ell_0}{1-\gamma}.$$

Proposition

Assume either $\gamma = 1$ *or* $\gamma < 1$ *and Condition C. Then, for any* $n \leq N$*,*

$$V_n^-(x) = V^-(x), \quad \textit{for all } x \in \mathrm{Acc}_n,$$

and the algorithm converges in $N \leq |\mathcal{X} \setminus \mathcal{X}_f|$ *steps to the value function* V^- *on the reachability set* $\mathcal{R}$*.*

Sketch of proof

From DPS, it suffices to prove that $V_1^-(x) = V^-(x)$. The inequality $V_1^-(x) \geq V^-(x)$ follows easily from the definitions. Now for

$$\bar{x} \in \mathrm{argmin}_{x \in \mathrm{Cons}_1} W_1(x)$$

consider an optimal pair $(\alpha^*, b^*_\bullet) \in \mathcal{A} \times B^{\mathbb{N}}$ and the corresponding optimal trajectory x_n starting from $\bar{x}$, that is,

$$\begin{aligned} x_{n+1} &= S(x_n, \alpha^*[b^*_\bullet]_n, b^*_n), \quad x_0 = \bar{x} \\ V^-(\bar{x}) &= J(\bar{x}, \alpha^*[b^*_\bullet], b^*_\bullet). \end{aligned}$$

If $\hat{n}(\bar{x}, \alpha^*[b^*_\bullet], b^*_\bullet) = 1$, then $V^-(\bar{x}) = W_1(\bar{x}) = V_1^-(\bar{x})$, which is the desired conclusion.

Sketch of proof

From DPS, it suffices to prove that $V_1^-(x) = V^-(x)$. The inequality $V_1^-(x) \geq V^-(x)$ follows easily from the definitions. Now for

$$\bar{x} \in \mathrm{argmin}_{x \in \mathrm{Cons}_1} W_1(x)$$

consider an optimal pair $(\alpha^*, b^*_\bullet) \in \mathcal{A} \times B^{\mathbb{N}}$ and the corresponding optimal trajectory x_n starting from $\bar{x}$, that is,

$$\begin{aligned} x_{n+1} &= S(x_n, \alpha^*[b^*_\bullet]_n, b^*_n), \quad x_0 = \bar{x} \\ V^-(\bar{x}) &= J(\bar{x}, \alpha^*[b^*_\bullet], b^*_\bullet). \end{aligned}$$

If $\hat{n}(\bar{x}, \alpha^*[b^*_\bullet], b^*_\bullet) = 1$, then $V^-(\bar{x}) = W_1(\bar{x}) = V_1^-(\bar{x})$, which is the desired conclusion.
If, instead, $\hat{n} := \hat{n}(\bar{x}, \alpha^*[b^*_\bullet], b^*_\bullet) > 1$ we will distinguish two cases.

- Case $\gamma = 1$. Since $\ell > 0$ we have that

$$V^-(\bar{x}) = \sum_{n=0}^{\hat{n}-2} \ell(x_n, \alpha^*[b_\bullet]_n, b_n^*) + V^-(x_{\hat{n}-1}) > V^-(x_{\hat{n}-1}).$$

On the other hand, we have an optimal pair strategy-control and corresponding optimal trajectory starting from $x_{\hat{n}-1}$ that reaches $\mathcal{X}_f$ in one step. Then $V^-(x_{\hat{n}-1}) = W_1(x_{\hat{n}-1})$ and so

$$V^-(x_{\hat{n}-1}) = W_1(x_{\hat{n}-1}) \geq W_1(\bar{x}) = V_1^-(\bar{x}) \geq V^-(\bar{x})$$

which is a contradiction.

- Case $\gamma = 1$. Since $\ell > 0$ we have that

$$V^-(\bar{x}) = \sum_{n=0}^{\hat{n}-2} \ell(x_n, \alpha^*[b_\bullet]_n, b_n^*) + V^-(x_{\hat{n}-1}) > V^-(x_{\hat{n}-1}).$$

On the other hand, we have an optimal pair strategy-control and corresponding optimal trajectory starting from $x_{\hat{n}-1}$ that reaches $\mathcal{X}_f$ in one step. Then $V^-(x_{\hat{n}-1}) = W_1(x_{\hat{n}-1})$ and so

$$V^-(x_{\hat{n}-1}) = W_1(x_{\hat{n}-1}) \geq W_1(\bar{x}) = V_1^-(\bar{x}) \geq V^-(\bar{x})$$

which is a contradiction.

- Case $\gamma < 1$. Follow the same argument and use Condition C in the last part, to show that for player 1 it is always more convenient to follow a path with a smaller number of steps.

Some remarks

- The algorithm has the computational advantages as Dijkstra. In particular, for constant costs l and g, all considered nodes are accepted and hence the value function is computed only once in each node, i.e., the algorithm is single pass.
- If $\gamma = 1$, $\ell \equiv 1$, and $g \equiv 0$, the problem for player 1 is the shortest length of paths reaching $\mathcal{X}_f$ while player 2 maximizes such length. If in addition B is a singleton, the problem reduces to the classical shortest path and the algorithm is exactly Dijkstra.
- If $\gamma = 1$, we can add a final step to the algorithm by setting $V^-_{N+1}(x) := W_0(x) = +\infty$ for all $x \in \mathcal{X} \setminus \mathrm{Acc}_N$, so $V^-_{N+1}(x) = V^-(x)$ for $x \in \mathcal{X} \setminus \mathcal{R}$ and we have convergence on the whole state space $\mathcal{X}$.

More questions

- Question 1B: is there a Dijkstra-type algorithm for stochastic games?

  It is open, some extensions for a single player and stochastic transitions are in Bertsekas (book 2001) Vladimirsky (MOR 2008).

More questions

- Question 1B: is there a Dijkstra-type algorithm for stochastic games?

  It is open, some extensions for a single player and stochastic transitions are in Bertsekas (book 2001) Vladimirsky (MOR 2008).

- Question 2: can we use our algorithm for the discrete schemes associated to differential games?

  We discuss it in the next slides, but in general there are troubles, even for 1 player, see Cacace, Cristiani and Falcone (SIAM J. Sci. Comp. 2014)

Differential games

Consider a continuous-time dynamical system controlled by two players

$$y'(t) = f(y(t), a(t), b(t)), \quad y(0) = x.$$

We are given a closed target $\mathcal{T} \subseteq \mathbb{R}^n$, and consider the first time the trajectory $y_x(\cdot; a, b)$ hits $\mathcal{T}$

$$t_x(a, b) := \inf\{t : y_x(t; a, b) \in \mathcal{T}\},$$

and the the cost functional (for player 1)

$$\tilde{J}(x, a, b) := \int_0^{t_x} l(y(t), a(t), b(t)) e^{-\lambda t} dt + e^{-\lambda t_x} g(y_x(t_x; a, b)),$$

for measurable controls $a \in \tilde{\mathcal{A}}$, $b \in \tilde{\mathcal{B}}$, $\lambda \geq 0$ is the discount rate.

Call Γ the set of non-anticipating strategies for player 1.

Hamilton-Jacobi-Isaacs equation

The lower value of the game (Varaiya, Roxin, Eliott-Kalton) is

$$v^-(x) := \inf_{\alpha\in\Gamma} \sup_{b\in\tilde{\mathcal{B}}} \tilde{J}(x, \alpha[b], b).$$

Under natural conditions it is a viscosity solution of the HJI equation

$$\lambda v^- - \max_{b\in B} \min_{a\in A} \left\{ f(x,a,b)\cdot Dv^- + l(x,a,b) \right\} = 0 \text{ in } \Omega := \mathbb{R}^d \setminus \mathcal{T}$$

with the boundary condition $\quad v^-(x) = g(x)\ \forall\, x \in \partial\mathcal{T}$.

Hamilton-Jacobi-Isaacs equation

The lower value of the game (Varaiya, Roxin, Eliott-Kalton) is

$$v^-(x) := \inf_{\alpha\in\Gamma} \sup_{b\in\tilde{\mathcal{B}}} \tilde{J}(x, \alpha[b], b).$$

Under natural conditions it is a viscosity solution of the HJI equation

$$\lambda v^- - \max_{b\in B} \min_{a\in A} \left\{ f(x,a,b)\cdot Dv^- + l(x,a,b) \right\} = 0 \ \text{ in } \Omega := \mathbb{R}^d \setminus \mathcal{T}$$

with the boundary condition $\quad v^-(x) = g(x) \; \forall\, x \in \partial\mathcal{T}$.

For a time step $h > 0$ consider the discrete-time game with

$$S(x,a,b) = x + hf(x,a,b), \quad \ell(x,a,b) = hl(x,a,b), \quad \gamma = e^{-\lambda h},$$

a natural approximation of the differential game.

Take also a finite grid $\mathcal{X}$ with final nodes $\mathcal{X}_f := \mathcal{X} \cap \mathcal{T}$.

Discrete (Hamilton-Jacobi-)Isaacs equation

The Discrete (HJ)I equation, semi-Lagrangian approximation of the HJI PDE, is

$$W(x) = \max_{b \in B} \min_{a \in A} \{\ell(x, a, b) + \gamma W(S(x, a, b))\}, \quad \forall x \in \mathcal{X} \setminus \mathcal{X}_f,$$

with the boundary condition $W(x) = g(x), \quad \forall x \in \mathcal{X}_f$.
In general $S(x, a, b) \notin \mathcal{X}$, so in the right hand side W must be extended by interpolation among the neighbouring nodes.
Call $k =$ mesh size of the grid, $W_{h,k} =$ solution of DI equation + BC.

Theorem (M.B. - Falcone - Soravia 94)

If $k/h \to 0$ *the weak (viscosity) semilimits*

$$\overline{W}(x) := \limsup_{h,k \to 0,\, y \to x} W_{h,k}(y), \quad \underline{W}(x) := \liminf_{h,k \to 0,\, y \to x} W_{h,k}(y)$$

are a sub- and a supersolution of the HJI PDE.

The weak convergence above becomes local uniform convergence of $W_{h,k} \to v^-$ if the HJI PDE + BC has a continuous solution, by the Comparison Principle.

- Question: Can we combine this convergence result with a Dijkstra-type algorithm?

The weak convergence above becomes local uniform convergence of $W_{h,k} \to v^-$ if the HJI PDE + BC has a continuous solution, by the Comparison Principle.

- Question: Can we combine this convergence result with a Dijkstra-type algorithm?

In general this is not obvious and there are indeed troubles, even for a single player: see Cacace, Cristiani, Falcone (SIAM J. Sci. Comp. 2014). A simple positive case: grid adapted to the dynamics, i.e.,

$$S(x,a,b) \in \mathcal{X} \quad \forall\, x \in \mathcal{X} \setminus \mathcal{X}_f,\ a \in A,\ b \in B. \qquad \text{(AG)}$$

Then $W = W_{h,k}$ in the DI equation can be computed only on the nodes, without any interpolation procedure.

Proposition

Under the assumption (AG) the solution W of the Discrete Isaacs equation coincides with the lower value function V^- of the discrete game. Thus it can be computed by the Dijkstra-type algorithm.

Grids adapted to the dynamics

Example 1. If $f = f(a,b)$ independent of x, can build an adapted grid by

$$\mathcal{X}_0 := \mathcal{X}_f, \qquad \mathcal{X}_{n+1} := \{x - hf(a,b) \text{ for some } x \in \mathcal{X}_n,\ a \in A,\ b \in B\}.$$

Example 2. For the *convex-concave eikonal equation* in the rectangle $\Omega = (0,c) \times (0,d) \subseteq \mathbb{R}^2$

$$|u_x| - \delta|u_y| = l(x,y) \text{ in } \Omega, \qquad u(x,y) = g(x,y) \text{ on } \partial\Omega,$$

the associated differential game has dynamics

$$x' = a,\ y' = b, \quad a \in \{-1,1\},\ b \in \{-\delta,\delta\}.$$

A rectangular grid $\mathcal{X} = \{(jh, k\delta h) : j = 1,\dots,\frac{c}{h},\ k = 1,\dots,\frac{d}{\delta h}\}$ is adapted to the dynamics for all $h = h_n$ of a sequence $h_n \to 0$ if $c\delta/d$ is rational.

Trouble: for adapted grids $k = O(h)$ instead of $k/h \to 0$, so cannot apply the BFS theorem.

Convergence on admissible sequences of grids

For $h_n \to 0$ take a sequence of grids $\mathcal{X}^n$ adapted to the dynamics with time step h_n and such that

$$\forall\, x \in \mathbb{R}^d \quad \exists\, x^{(n)} \in \mathcal{X}^n \text{ such that } \lim_n x^{(n)} = x,$$

This is an *admissible sequence of grids*. For each pair $h_n, \mathcal{X}^n$ call W_n the solution of the Discrete Isaacs equation and define the weak semi-limits

$$\overline{W}(x) := \limsup_{\mathcal{X}^n \ni x^{(n)} \to x} W_n(x^{(n)}), \quad \underline{W}(x) := \liminf_{\mathcal{X}^n \ni x^{(n)} \to x} W_n(x^{(n)}), \quad x \in \overline{\Omega}.$$

Proposition

Assume h_n, $\mathcal{X}^n$, W_n are as above with W_n locally equibounded.
Then $\overline{W}$ and $\underline{W}$ are, respectively, a viscosity sub- and supersolution of the H-J-Isaacs PDE.

As before if $v^- \in C(\overline{\Omega})$ then $\overline{W} = \underline{W} = v^-$ by the Comparison Principle. This implies the following form of *uniform convergence of W_n to v^-*:

for all $\epsilon > 0$ and compact set K there exists $\overline{n}$ and $\delta > 0$ such that

$$|W_n(x^{(n)}) - v^-(x)| < \epsilon \quad \forall\, x \in K,\ x^{(n)} \in \mathcal{X}^n,\ n \geq \overline{n},\ |x^{(n)} - x| < \delta.$$

Rmk: no condition on k/h in the last result!

How were we in 1994?

How were we in 1994?

Look rather serious ... must go to the beach!

Thanks for your attention and

Happy Birthday Maurizio!