About upper bounds on the complexity of Policy Iteration*

Romain Hollanders, Balázs Gerencsér, Jean-Charles Delvenne, Raphaël M. Jungers†
Department of Mathematical Engineering and ICTEAM, UCLouvain, Belgium
E-mail: firstname.name@uclouvain.be.

Abstract

We consider Acyclic Unique Sink Orientations of the n-dimensional hyper-cube (AUSOs), that is, acyclic orientations of the edges of the hyper-cube such that any sub-cube has a unique vertex of maximal in-degree. We study the Policy Iteration (PI) algorithm, also known as Bottom-Antipodal or Switch-All, to find the global sink: starting from an initial vertex $\pi_0, i = 0$, the outgoing links at the present vertex π_i define a sub-cube of the AUSO. Policy Iteration jumps to the vertex π_{i+1} that is antipodal to π_i in this sub-cube. This procedure is repeated until the sink is found. Finite-time convergence is guaranteed. Policy Iteration was shown to require at most $6 \cdot \frac{2^n}{n}$ steps to converge by Mansour & Singh [MS99]. Our goal in this work is to improve this bound.

Our first contribution is to provide the first improvement over Mansour & Singh's bound after fifteen years, namely a $(2 + o(1))\frac{2^n}{n}$ upper bound (Section 2). Perhaps more importantly, we also show in Section 3 that this bound is optimal for an important relaxation of the problem.

Policy Iteration was originally designed to solve Markov Decision Processes (MDPs) for which our new bound also holds. The algorithm and the bound also apply to 2-Player Turn-Based Stochastic Games (2TBSGs), a two player generalization of MDPs for which no polynomial-time algorithms are known. In MDPs and 2TBSGs, the vertices of an AUSO correspond to policies or strategies and the goal is to find the policy that optimizes some objective function and that corresponds to the global sink. The partial order that shapes the policies of an MDP or a 2TBSG is an AUSO with some additional properties. The best known lower bounds for the number of steps of PI are given by $\Omega\big(\sqrt[7]{2}^n\big)$ for MDPs [Fea10], by $\Omega\big(\sqrt[9]{2}^n\big)$ for Parity Games, a special case of 2TBSGs [Fri09] and by $\Omega\big(\sqrt{2}^n\big)$ for AUSOs [SS05]. Establishing these bounds was a major milestone for the study of the Simplex algorithm for Linear Programming as they lead to exponential lower bounds for some critical pivoting rules [Fri11, FHZ11]. More details are given in Section 5.

Our second contribution concerns an alternate relaxation of the upper bound problem proposed by Hansen & Zwick [Han12] that can essentially be formulated as follows. Let $A \in \{0,1\}^{m \times n}$ be a binary matrix such that for every pair of rows i, j of A with $1 \leq i < j \leq m$, there exists a column k such that:

$$A_{i,k} \neq A_{i+1,k} = A_{j,k} = A_{j+1,k}.$$

The maximum number of rows of such a matrix A for a given number of columns n is also a bound for the number of steps of PI for an n-dimensional cube. Using exhaustive search for the number of columns of A ranging from 1 to 7, Hansen & Zwick obtained $2, 3, 5, 8, 13, 21, \geq 33$ as its maximum number of rows. Given these observations, they conjectured that the maximum number of steps of PI should be given by F_{n+2}, the $(n+2)^{\text{nd}}$ Fibonacci number. From the numerical results, their conjecture appears like a perfect match, but confirming it for $n = 7$ has been claimed as an interesting, yet computationally hard challenge. It was proposed as January 2014's IBM *Ponder This* challenge.

As our second contribution, we show Hansen & Zwick's conjecture wrong. In Section 4 we explain how we solved the IBM challenge by searching through every 7-column candidate matrix with finally nothing better than 33 rows.

*A preliminary version of this work has been presented at the 25th International Conference on Probabilistic, Combinatorial and Asymptotic Methods for the Analysis of Algorithms (AofA'2014).

†This work was supported by an ARC grant from the French Community of Belgium and by the IAP network 'Dysco' funded by the office of the Prime Minister of Belgium. The scientific responsiblity rests with the authors. J.-C. D. is with CORE and NAXYS. R. M. J. is an F.R.S./FNRS Research Associate.

1 The complexity of Policy Iteration: problem properties and relaxation

In this section, we first formalize the problem of bounding the complexity of Policy Iteration. We then formulate the key properties that will allow us to derive our new upper bound. We then close the section by formulating an important relaxation of the problem for which our bound is optimal.

1.1 Problem statement

Definition 1 (Partial ordering)**.** An *Acyclic Unique Sink Orientation* of a cube (AUSO) defines a *partial ordering* $\prec$ on the vertices of the n-dimensional cube $\{0,1\}^n$ such that:

(1) For any *neighbors* π and $\pi' \in \{0,1\}^n$ (i.e., that differ in exactly one entry), either $\pi \prec \pi'$ if the connecting edge is oriented from π to π' or $\pi \succ \pi'$ otherwise.

(2) The partial order is acyclic;

(3) Any sub-cube of $\{0,1\}^n$ has a unique *sink*, i.e. a unique vertex with maximal in-degree.

We write $\pi \prec \pi'$ if and only if there exists a directed path from neighbor to neighbor connecting π to π'. In that case, we say that π' *dominates* π. An example AUSO for $n = 3$ is given in Figure 1A.

Definition 2 (Switching)**.** Let $\pi \in \{0,1\}^n$ and $U \subseteq \{1,\dots,n\}$. We define $\pi' \triangleq \pi \oplus U$ as the vertex obtained from π by switching all the bits corresponding to the entries in U from 0 to 1 or from 1 to 0. In the definition, we call $\oplus$ the *switching* operator. Note that U can be represented by a vector of $\{0,1\}^n$ where entry i is 1 if $i \in U$, 0 otherwise and the switching operator can then be seen as an entry-wise *xor* between π and U.

Definition 3 (Improvement set)**.** Given a vertex π, we define the *Improvement set* of π as its set of improving directions:

$$T^\pi = \big\{ i \ : \ \pi \oplus \{i\} \succ \pi \big\}.$$

Using the above notations, the *Policy Iteration* algorithm is described by Algorithm 1. A possible run is illustrated by Figure 1B.

Initialization: π_0, $i = 0$
while $T^{\pi_i} \neq \emptyset$ **do**
  $\pi_{i+1} = \pi_i \oplus T^{\pi_i}$
  $i \leftarrow i + 1$
end
return π_i

Algorithm 1: (Greedy) Policy Iteration

The following two propositions are the key results that guarantee the finite-time convergence of PI. More precisely, Proposition 1 guarantees that every iterate of PI dominates the previous ones so that we never visit the same vertex twice and Proposition 2 provides the stopping criterion.

Proposition 1 (Theorem 1 in [MS99])**.** *For any $U \subseteq T^\pi$, $U \neq \emptyset$, we have $\pi \prec \pi \oplus U$.*

Proposition 2 (Theorem 2 in [MS99])**.** *For any given π, if $T^\pi = \emptyset$, then π is the global sink.*

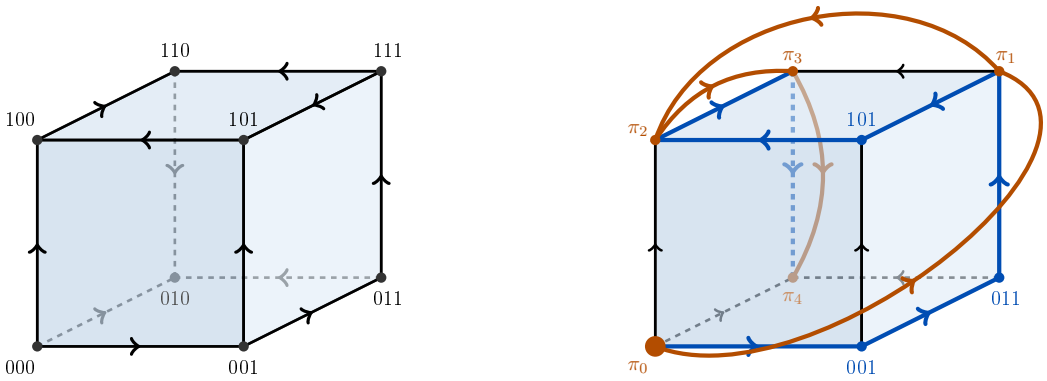

Figure 1: (Left) An example of an AUSO. Here, vertex 010 is the global sink. (Right) An example of a PI run on this worst case AUSO for dimension 3. Here PI starts from vertex 000. At each step, it jumps (with the red arrows) from the current vertex to its antipodal vertex in the sub-cube spanned by the outgoing edges of the current vertex. It stops when it reaches the global sink. The red vertices make the PI-sequence and the blue path represents a supersequence of the PI-sequence.

The following definition helps us to properly define our goal.

Definition 4 (PI-sequence)**.** We refer to the policies $\pi_0 \prec \cdots \prec \pi_{m-1}$ explored by greedy PI as a *PI-sequence*. Here m is the length of the PI-sequence.

Our ultimate goal then becomes the following.

Problem 1. Find the longest possible PI-sequence.

1.2 Main properties

The argument behind the following Lemma was already known by Mansour & Singh, even though they only formulated a corollary of it.

Lemma 3 (Lemma 6 in [MS99])**.** *For any two vertices $\pi \prec \pi'$ such that $\pi' = \pi \oplus T^\pi$, $|T^\pi| = k$, there exists a sequence of policies $\pi^{(0)}, ..., \pi^{(k)}$ such that:*

$$\pi = \pi^{(0)} \prec \pi^{(1)} \prec ... \prec \pi^{(k)} = \pi',$$

and such that $\pi^{(i)}$ and $\pi^{(i+1)}$ are neighbors for all $0 \leq i < k$.

Definition 5 (Subsequence and supersequence)**.** Let O be a total ordering. We call *subsequence* of O any ordered subset of elements of O. We call *supersequence* of O any total ordering that contains O as a subsequence.

Lemma 3 indicates that any PI-sequence allows an underlying supersequence that corresponds to a path in an AUSO, as illustrated by the blue path in Figure 1B. In [MS99], Mansour & Singh formulated the following corollary which is perhaps the most interesting consequence of Lemma 3.

Corollary 4 (Jumping)**.** *Let π_i and π_{i+1} be two vertices of a PI-sequence. Then, there exists a supersequence of the PI-sequence with at least $|T^{\pi_i}|$ different vertices between π_i and π_{i+1}, that is, $|T^{\pi_i}|$ vertices π such that $\pi_i \prec \pi \prec \pi_{i+1}$. When we step from π_i to π_{i+1}, we say that we* jump *$|T^{\pi_i}|$ policies of the supersequence.*

Corollary 4 indicates that if the current vertex at any PI step has a large improvement set, then many vertices are jumped. Intuitively, the smaller the improvement set of the policies explored by PI, the more iterations we can expect.

Together with Corollary 4, the following property is a key to our results. It is an extension of Lemma 5 in [MS99] to any pair of vertices, not only vertices of a PI-sequence.

Proposition 5 (Non-inclusion, see Lemma 5 in [MS99]). *If $\pi \prec \pi'$, then $T^\pi \not\subseteq T^{\pi'}$.*

One of the implications of Proposition 5 is that the improvement sets of all the vertices of a total order are distinct. Therefore, a total ordering of vertices, like any supersequence of a PI-sequence, translates to the total ordering of their (distinct) improvement sets. For two vertices π, π' of this total ordering, the order $T^\pi \prec T^{\pi'}$ is thus properly defined by $\pi \prec \pi'$. Similarly, we can refer to the improvement sets of a PI-sequence by $T_0, ..., T_{m-1}$ where m is the number of iterations of PI and T_i is a shortcut for T^{π_i}. Figure 2 illustrates the translation of the vertices of the blue path from Figure 1B to the ordering of their improvement sets.

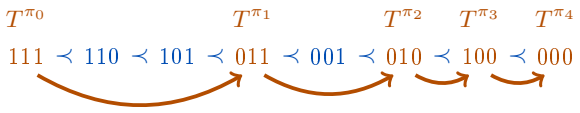

Figure 2: A supersequence of the PI-sequence from Figure 1B, here expressed in terms of the improvement sets of the policies, i.e., their sets of outgoing edges. Observe that the supersequence satisfies Proposition 5 and the PI-sequence (in red) satisfies Corollary 4 on this supersequence.

Note that Mansour & Singh were not able to take full advantage of Proposition 5 to derive their bounds: they only used the fact that the improvement sets of a PI-sequence should be distinct but they did not use the non-inclusion part, neither did they exploit our extension to any pair of vertices. They therefore mentioned these ideas as possible keys to further improve their upper bound. In Section 3, we show the limitation of this approach.

1.3 An important relaxation

Interestingly, Proposition 5 and Corollary 4 are the only properties that we need about PI to derive our upper bound in the next section. In accordance with this observation, we relax Problem 1 in such a way that our arguments still hold. We will then show in Section 3 that our bound is optimal for the relaxation.

Definition 6 (Pseudo-PI-sequence). We call *pseudo-PI-sequence* of size m a triple of the following.

- A sequence $\Pi = \pi_0, \pi_1, \ldots, \pi_{m-1}$ of vertices. We define the abstract ordering $\prec$ on the sequence Π by the ordering of their indices.
- A total ordering O on some vertices of $\{0,1\}^n$ which is a supersequence of Π.
- An abstract improvement set T^π for every vertex π appearing in O.

We require the claim from Proposition 5 to hold for O and we require Π to satisfy Corollary 4 as a subsequence of O.

Definition 6 is a simplification of the original PI-sequence mainly in the sense that it forgets about some of the underlying AUSO structure that carries it. Note that there is a natural way of constructing a pseudo-PI-sequence from any PI-sequence. Proposition 5 and Corollary 4 still hold for pseudo-PI-sequences by design. Problem 1 is then relaxed as follows.

Relaxation 1. Find the longest possible pseudo-PI-sequence.

2 Improving the upper bound on PI complexity

The following Theorem provides a factor 3 improvement over Mansour & Singh's bound. The result is obtained from Mansour & Singh's original arguments coupled with tighter probability estimates.

Theorem 1. *The number of steps of Policy Iteration is bounded above by* $\big(2 + o(1)\big) \cdot \frac{2^n}{n}$.

Proof. The proof proceeds in two steps. First, we consider "small" improvement sets and show that there are at most $o\left(\frac{2^n}{n}\right)$ of them in a PI-sequence. Then we consider "large" improvement sets and show that PI explores at most $2 \cdot \frac{2^n}{n} + o\left(\frac{2^n}{n}\right)$ of them because they jump many vertices on the way.

Small improvement sets. We consider the small improvement sets T_i such that $|T_i| \leq \frac{n}{2} - f(n)$ with

$$f(n) \triangleq \sqrt{n \log n}.$$

From Proposition 5, the same improvement set can appear at most once in the PI-sequence, hence the maximum number of small improvement sets can be expressed in a probabilistic fashion:

$$\begin{aligned}\sum_{d=0}^{\lfloor \frac{n}{2} - f(n) \rfloor} \binom{n}{d} &= 2^n \sum_{d=0}^{\lfloor \frac{n}{2} - f(n) \rfloor} \binom{n}{d} \left(\frac{1}{2}\right)^d \left(\frac{1}{2}\right)^{n-d}, \\ &= 2^n \cdot P\left[X \leq \frac{n}{2} - f(n)\right],\end{aligned}$$

where $X \sim \text{Bin}\left(n, \frac{1}{2}\right)$ follows a binomial distribution. Using Hoeffding's inequality, we have:

$$P\left[X \leq n \cdot \left(\frac{1}{2} - \frac{f(n)}{n}\right)\right] \leq e^{-2 \cdot \left(\frac{f(n)}{n}\right)^2 \cdot n} = \frac{1}{n^2}$$

Therefore we have:

$$\sum_{d=0}^{\lfloor \frac{n}{2} - f(n) \rfloor} \binom{n}{d} \leq 2^n \cdot \frac{1}{n^2} = o\left(\frac{2^n}{n}\right).$$

Large improvement sets. We now consider the improvement sets T_i such that $|T_i| > \frac{n}{2} - f(n)$ and show that these sets jump many policies on the way and hence that we cannot encounter many of them in a PI-sequence. Suppose that we encounter k such improvement sets. Then, from Corollary 4, we jump at least $k \cdot (\frac{n}{2} - f(n))$ distinct policies. Since we cannot jump more that 2^n policies, we have the following condition on k:

$$k \leq \frac{2^n}{\frac{n}{2} - f(n)} = 2 \cdot \frac{2^n}{n} \cdot \frac{1}{1 - \sqrt{\frac{4 \log n}{n}}}.$$

Hence $k \leq 2 \cdot \frac{2^n}{n} + o\left(\frac{2^n}{n}\right)$. □

3 A relaxation where our bound is optimal

The following theorem shows that the upper bound from Theorem 1 is tight for Relaxation 1.

Theorem 2. *There exists a pseudo-PI-sequence of size* $2 \cdot \frac{2^n}{n} + o\left(\frac{2^n}{n}\right)$.

Proof. Let us define $\mathcal{T}^d$ as the set of all improvement sets of cardinality d. Then, let us sort the 2^n distinct sets in decreasing order of cardinality such that $T \in \mathcal{T}^d$ comes before $T' \in \mathcal{T}^{d'}$ if $d > d'$. Any total ordering of the improvement sets of given cardinality can be chosen. This total ordering clearly satisfies the claim of Proposition 5. We then select a subsequence Π from this ordering in a greedy way so as to ensure Corollary 4: starting with the first improvement set T_0 (which has cardinality n according to our construction), at each step i, we "jump" $|T_i|$ elements in the ordering to select T_{i+1}. This procedure is illustrated by Figure 3. There are $\binom{n}{d}$ sets in $\mathcal{T}^d$ that are all gathered together in the ordering. Therefore, following the greedy choice of elements, we pick at least $\frac{1}{d+1}\binom{n}{d}$ sets from $\mathcal{T}^d$ if $d < n$, for a total number of steps of at least:

$$
\begin{aligned}
&1 + \sum_{d=0}^{n-1} \frac{1}{d+1} \binom{n}{d} \\
&= 1 + \frac{1}{n+1} \sum_{d=0}^{n-1} \frac{(n+1)!}{(d+1)! \cdot \big((n+1)-(d+1)\big)!} \\
&\geq \frac{1}{n+1} \sum_{d=-1}^{n} \binom{n+1}{d+1} \\
&= \left(1 - \frac{1}{n+1}\right) \frac{2^{n+1}}{n}.
\end{aligned}
$$

□

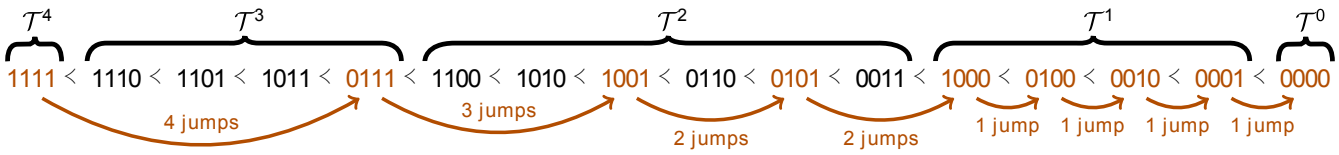

Figure 3: For our lower bound, the total ordering O from the pseudo-PI-sequence is defined as the decreasing cardinality ordering of the improvement sets. Here it is shown for $n = 4$. The claim of Proposition 5 is satisfied on this ordering. The improvement sets that appear in red correspond to the sequence Π from Definition 6 and they ensure Corollary 4 with the supersequence O. In this setting, there are at least $\big(2 + o(1)\big) \cdot \frac{2^n}{n}$ elements in Π, as stated by Theorem 2.

Theorem 2 tends to show that conditions acting only on the improvement sets of the vertices are not sufficient to obtain better bounds. Instead, it seems to be essential to consider more of the AUSO structure of the partial order. This is the goal of the next section, where we explore a different relaxation of Problem 1.

4 The Fibonacci conjecture for PI fails

To relax Problem 1, Hansen & Zwick suggested the following idea. Let $A \in \{0,1\}^{m \times n}$ be a binary matrix whose $(i+1)^{\text{th}}$ row corresponds to the vertex π_i in a PI-sequence, written as a binary row vector. Hansen & Zwick showed that such a matrix that comes from an actual PI-sequence must satisfy the *order-regularity* condition: for every rows i, j of A with $1 \leq i < j \leq m$, there must exist a column k such that:

$$A_{i,k} \neq A_{i+1,k} = A_{j,k} = A_{j+1,k}, \tag{1}$$

where $A_{m+1,k} \triangleq A_{m,k}$ for all $1 \leq k \leq n$ when $j + 1 > m$. Furthermore, the last two rows (labeled $m-1$ and m) are required to be distinct. Given this condition, any bound on the number of rows m of A given its number of columns n is also a bound for the number of steps of PI. As mentioned in the introduction, Hansen & Zwick conjectured the following based on empirical experiments.

Conjecture 1 (Hansen & Zwick, 2012)**.** *The maximum number of steps of PI is given by F_{n+2}, the $(n+2)^{nd}$ Fibonacci number.*

0	0	0
1	1	1
0	0	1
0	1	1
0	1	0

0	0	0	0
1	1	1	1
0	0	0	1
0	1	1	1
0	0	1	0
0	1	1	0
0	1	0	0
1	1	0	0

Table 1: The unique extremal order-regular matrices for 3 and 4 columns

Hansen & Zwick's order-regularity condition can be viewed as the translation of the AUSO features, but only on the vertices of the PI-sequence (it disregards all the other vertices that are jumped during the iterative process but that still exist in the cube). Hence solving Problem 1 on order-regular matrices is a relaxation of solving the same problem on AUSOs.

Our aim is to improve the computational methods leading to Hansen & Zwick's conjecture. They already performed an exhaustive search for order-regular matrices up to $n = 6$ columns resulting in the first few elements of the Fibonacci sequence. Moreover, they generated some 33×7 matrices using a non-exhaustive search for sequences of dimension 7. We enhance the computational approach to a level where exhaustive search is possible for dimension 7 and thus we show that 34×7 order-regular matrices do not exist.

Let us first illustrate the complexity of the problem. The expected number of rows is exponential in the number of columns, and the number of possible binary matrices is exponential in the number of elements in the matrix, thus the search space is doubly exponential in the input n. For instance, we should check 2^{238} matrices for $n = 7$, against 2^{126} for $n = 6$. Obviously this has to be radically decreased to be computationally feasible. For another way of emphasizing the difficulty of stepping from $n = 6$ to $n = 7$, note that our final code took 1 month to run for $n = 7$ while it finished in 10 seconds for $n = 6$ (using 10 Intel® Xeon® X5670 cores). Next we present the techniques used to reduce the computational time.

Symmetry. The following elementary transformations do not change whether a matrix is order-regular or not. In any column we may swap 0s and 1s. We can also apply a permutation to the columns. Therefore we can assume that the first row is all 0 and that columns follow each other in a lexicographical order. This way we remove redundancy in the search space.

Branching. If the first block of d rows of a matrix is infeasible itself there is no need to check how the rest of the matrix looks like. On the other hand, if the first d rows of several matrices are the same, it is unnecessary to recheck this part every time. We exploit these observations by using a depth first search on the matrices. If we have an initial block of d rows that is feasible, we try every extension to $d+1$ rows

0	0	0	0	0	0	0
1	1	1	1	1	1	1
0	0	0	0	0	0	1
0	1	1	1	1	1	1
0	0	0	0	0	1	0
1	0	1	1	1	1	1
0	0	0	0	1	1	0
1	0	0	1	1	0	1
0	1	0	0	1	1	1
1	1	0	1	0	0	1
0	1	0	1	1	1	0
1	1	0	1	1	0	1
0	1	0	0	1	0	0
1	1	1	0	1	0	1
1	0	0	0	1	1	0
1	1	1	0	0	1	1
1	0	0	0	0	1	0
1	1	1	1	0	1	0
1	1	0	0	0	0	0
1	1	1	0	1	1	0
1	1	0	0	1	0	0
1	1	1	1	1	0	0
0	1	1	0	0	0	0
1	0	1	0	1	0	0
0	1	1	0	1	0	0
0	0	1	1	1	0	0
0	0	1	0	0	0	1
0	0	1	0	1	0	0
0	0	1	0	1	0	1
0	0	1	1	1	0	1
0	0	0	1	1	0	1
0	0	0	1	1	1	1
0	1	0	1	1	1	1

Table 2: An example of a maximal 33×7 order-regular matrix

and only continue with those that do not violate the order-regularity condition. We are thus exploring a huge search tree.

Filtering. During the branch search, assume we are investigating a branch with the first d rows fixed. In any extension, any pair of rows has to be compatible with the same first d rows. We can see these pairs as the rows labeled j and $j+1$ in the order-regularity condition, to be compared with the pairs labeled i and $i+1$ with $i < d$. When extending a branch, pairs can only become incompatible with the already fixed rows. Therefore we filter the possible pairs of rows (π_1, π_2) that are compatible when we start checking the branch, and use only extensions that come from this filtered subset. We then prune this subset as we go down in the search tree. That way, at the cost of a reasonable amount of memory, we never have to check the same quadruplet of rows more than once in a given branch.

Direct cutting. The fact of having access to the set of filtered pairs of rows has an additional advantage. Assume we are looking at a branch with d rows fixed and we have r distinct rows appearing as π_1 in the filtered set of pairs of rows. In this case there is clearly no way of getting more than $d + r$ rows by extending this particular branch. Consequently if $d + r < 33$ then we discard the branch right away. This way we still retain every matrix that has 33 or more rows and we only need to search a restricted portion of the whole search tree.

Parallelization. Finally, we split the search space into smaller parts and develop a code that is capable of parallel processing.

Employing each of the above ideas contributes to the speedup of the algorithm. By using all of them, we succeeded in pushing the total runtime below one month, which is in the viable range for humans. In the end, we found several matrices of size 33×7 (more precisely 35 matrices for which the start block of 9 rows is different), but none of size 34×7. Therefore the true upper limit for the size of order-regular matrices deviates from the Fibonacci series.

5 The problem originates from Markov Decision Processes

The question of Problem 1 originates from the framework of Markov Decision Processes (MDPs) for which Policy Iteration was first defined by Howard in 1960 [How60]. MDPs are described from a set of n states in which a system can be. When being in a state, the system must choose an available action in that state, each of which induces a cost and moves the system to another state according to given transition probabilities. An MDP can always be reduced to a form in which every state has exactly two available actions, which is the case we restricted ourselves to in this work. A *policy* refers to the stationary choice of one action in every state. When fixing the action of every state according to the chosen policy, the MDP becomes a simple Markov chain. Given a policy, we can associate a value to each state of the MDP that corresponds to the infinite horizon expected cost of an agent starting in that state and following the policy thereafter. By solving an MDP, we mean providing the optimal policy that minimizes the value of every state. Such a policy always exists. Depending on the application, a total-, discounted- or average-cost criterion may be best suited to define the value function. Note that in every case, an optimal policy always exists. See e.g. [Ber07] and [Put94] for a comprehensive and in-depth study of MDPs and their cost criteria.

Several results exist on the complexity of PI for MDPs. Fearnley showed that PI requires at least $\Omega(\sqrt[7]{2}^n)$ steps to converge for total- and average-cost MDPs, a result that was later extended to general discounted-cost MDPs [HDJ12]. On the other hand, Ye showed that PI is strongly polynomial for discounted-cost MDPs with a fixed discount rate [Ye11]. Ye's bound was then further improved in [HMZ13, Sch13]. Post & Ye also showed that PI is strongly polynomial for Deterministic MDPs [PY13]. Their bounds were then again improved in [HKZ14]. The frontier between exponential and strong polynomial complexity is therefore narrow and characterizing MDP structures that belong to one or the other category is an interesting research topic.

Policy Iteration also applies to solve 2-Player Turn-Based Stochastic Games (2TBSGs), a generalization of MDPs to two players [Sha53, FV96]. In a 2TBSG, the set of states and their associated actions is partitioned among two players, the minimizer and the maximizer, that both have opposite objectives regarding the value function. When an action is played by a player, the associated cost is payed to the adversary, therefore the game is a zero-sum game. The rest of the dynamics is similar to that of an MDP and the optimal policy is reached when no player can change its strategy without deteriorating its payoff. An MDP can actually be seen as a 2TBSG where one player has no influence on the game.

Unfortunately to this day, no polynomial-time algorithms exist to solve general 2TBSGs. However, as for MDPs, Policy Iteration applied to 2TBSGs is strongly polynomial for the discounted-cost criterion when the discount factor is fixed [HMZ13]. On the other hand, Friedmann showed that PI applied to Parity Games, a special case of 2TBSGs for the average-cost criterion, may require $\Omega(\sqrt[9]{2}^n)$ steps to converge [Fri09]. He later extended his result in the context of the Simplex algorithm for Linear Programming and obtained exponential lower bounds for Zadeh's and for randomized pivoting rules, two major Simplex update rules that survived the attacks for decades [Fri11, FHZ11]. Note that Fearnley's example for MDPs was also built using Friedman's construction as model.

Acyclic Unique Sink Orientations were proposed as an accurate, yet relaxed representation of the MDP structure. The policies of an MDP are translated into the vertices of the hyper-cube and the values of the policies define a partial order among them which translates in the orientation of the edges of the hyper-cube. The resulting structure is acyclic and every sub-cube has a unique sink[1].

[1] This structure is even a bit richer than AUSOs as it also satisfies the Holt-Klee condition for instance [HK99, SS05]. A full characterization of this structure is still unknown today but AUSOs appear like a convenient relaxation.

6 Summary and perspectives

Relaxations are essential to obtain upper bounds on the number of iterations of Policy Iteration. In this work we first considered the relaxation through pseudo-PI-sequences and improved Mansour & Singh's bound to $\big(2+o(1)\big)\cdot\frac{2^n}{n}$. We then showed optimality of this bound for that relaxation. This revealed that more sophisticated properties are required for further improvements. We then considered the order-regularity condition as another relaxation of the AUSO structure for which Hansen & Zwick have proposed the exciting conjecture that the actual upper bound should be given by the Fibonacci numbers. However, we have proven this conjecture wrong. Still, the Fibonacci sequence could be a valid upper bound but then not a sharp one.

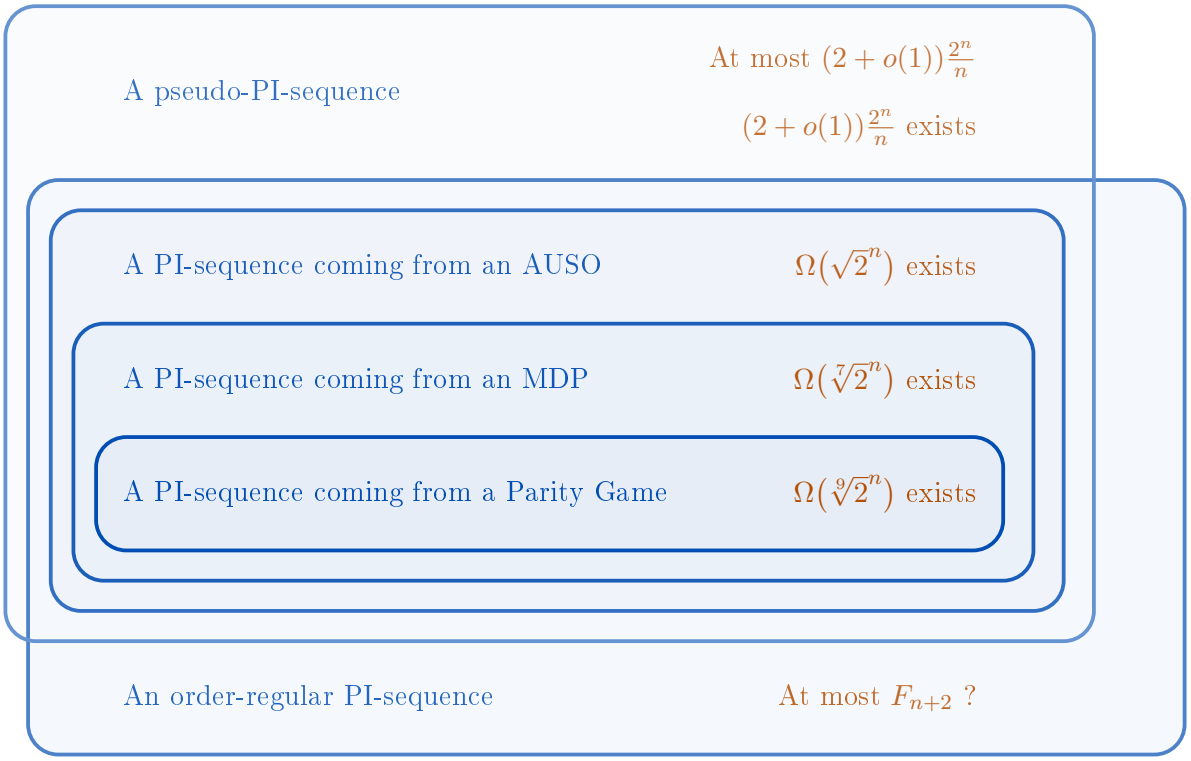

Figure 4: A schematic representation of the different known relaxations of the problem of bounding the complexity of Policy Iteration. The box A contains the box B if A is a relaxation of B. In that case, an upper bound (resp. a lower bound) that holds for box A (resp. for box B) also holds for box B (resp. for box A).

We graphically summarize the dependencies between the different formulations of the problem mentioned in this work in Figure 4. We also give an overview of the existing bounds that were obtained within each framework.

Clearly, there is still work to be done to reduce the gap between e.g. Fearnley's $\Omega\big(\sqrt[7]{2}^n\big)$ lower bound on the number of steps of PI for MDPs and our $\big(2+o(1)\big)\cdot\frac{2^n}{n}$ upper bound. For instance, it would be interesting to carry the $\Omega(\sqrt{2}^n)$ lower bound from Schurr & Szabó that holds for AUSOs [SS05] to MDPs. Another option would be to improve Fearnley's (or Friedmann's) constructions by reducing their number of states. In the end, we already know that the worst case of PI is exponential, but discovering the actual basis of the exponential remains a crucial question that will require additional hard work.

References

[Ber07] D. P. Bertsekas. *Dynamic Programming and Optimal Control.* Athena Scientific, Belmont, Massachusetts, 3rd edition, 2007.

[Fea10] J. Fearnley. Exponential Lower Bounds for Policy Iteration. *In Proceedings of the 37th International Colloquium on Automata, Languages and Programming (ICALP'10)*, pages 551–562, 2010.

[FHZ11] O. Friedmann, T. D. Hansen, and U. Zwick. Subexponential Lower Bounds for Randomized Pivoting Rules for the Simplex Algorithm. *Proceedings of the 43rd ACM Symposium on Theory of Computing, STOC*, 11, 2011.

[Fri09] O. Friedmann. An Exponential Lower Bound for the Parity Game Strategy Improvement Algorithm as we know it. *Proceedings of the 24th Annual IEEE Symposium on Logic In Computer Science*, pages 145–156, 2009.

[Fri11] O. Friedmann. A Subexponential Lower Bound for Zadeh's Pivoting Rule for Solving Linear Programs and Games. *Integer Programming and Combinatoral Optimization*, pages 192–206, 2011.

[FV96] J. Filar and K. Vrieze. *Competitive Markov decision processes.* Springer-Verlag New York, Inc., 1996.

[Han12] T. D. Hansen. *Worst-case Analysis of Strategy Iteration and the Simplex Method.* PhD thesis, Aarhus University, Science and Technology, Department of Computer Science, 2012.

[HDJ12] R. Hollanders, J.-C. Delvenne, and R. M. Jungers. The Complexity of Policy Iteration is Exponential for Discounted Markov Decision Processes. *In Proceedings of the 51st IEEE Conference on Decision and Control*, 2012.

[HK99] F. Holt and V. Klee. A proof of the strict monotone 4-step conjecture. *Contemporary Mathematics*, 223:201–216, 1999.

[HKZ14] T. D. Hansen, H. Kaplan, and U. Zwick. Dantzig's pivoting rule for shortest paths, deterministic mdps, and minimum cost to time ratio cycles. *In Proceedings of the 25th ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 847–860, 2014.

[HMZ13] T. D. Hansen, P. B. Miltersen, and U. Zwick. Strategy iteration is strongly polynomial for 2-player turn-based stochastic games with a constant discount factor. *Journal of the ACM (JACM)*, 60(1):1, 2013.

[How60] R. A. Howard. *Dynamic Programming and Markov Processes.* MIT Press, Cambridge, MA, 1960.

[MS99] Y. Mansour and S. Singh. On the Complexity of Policy Iteration. *In Proceedings of the 15th Conference on Uncertainty in Artificial Intelligence*, 1999.

[Put94] M. L. Puterman. *Markov Decision Processes.* John Wiley & Sons, 1994.

[PY13] I. Post and Y. Ye. The simplex method is strongly polynomial for deterministic Markov Decision Processes. *In Proceedings of the 24th ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 1465–1473, 2013.

[Sch13] B. Scherrer. Improved and generalized upper bounds on the complexity of Policy Iteration. *In Proceedings of the 27th Conference on Advances in Neural Information Processing Systems (NIPS)*, pages 386–394, 2013.

[Sha53] L. S. Shapley. Stochastic games. *Proceedings of the National Academy of Sciences of the United States of America*, 39(10):1095, 1953.

[SS05] I. Schurr and T. Szabó. Jumping doesn't help in abstract cubes. In *Integer Programming and Combinatorial Optimization*, pages 225–235. Springer, 2005.

[Ye11] Y. Ye. The Simplex and Policy-Iteration Methods are Strongly Polynomial for the Markov Decision Problem with a Fixed Discount Rate. *Mathematics of Operations Research*, 2011.