Likely-admissible and sub-symbolic heuristics

Marco Ernandes and Marco Gori
Dipartimento di Ingegneria dell'Informazione
Università di Siena (Italy)
{ernandes, marco}@dii.unisi.it

Abstract. It is well-known that while strict admissibility of heuristics in problem solving guarantees the optimality of the A* algorithm, many problems cannot be effectively faced because of the combinatorial explosion. In order to address this problem the notion of ϵ-admissible search has been introduced, which yields solutions with bounded costs [12].

In this paper, we introduce the related concept of likely-admissible heuristics, where the admissibility requirement is relaxed in a probabilistic sense. Instead of providing an upper-bound to the cost we guarantee to end up with optimal solutions with a given probability. Interestingly, likely-admissible heuristics can be obtained naturally by statistical learning techniques such as artificial neural networks, which can learn from examples the expected value of the cost to reach the target. We used multilayered neural networks with a proper novel cost function in order to bias the learning towards admissibility.

Our experiments with the 15-puzzle and IDA* show that the adoption of likely-admissible sub-symbolic heuristics yield optimal solutions in 50% of the cases, taking only 1/500 time (1/13000 space) of classic Manhattan-based search.

1 Introduction

The classical rigid interpretation of admissibility has driven the optimal search theory to a stalemate. Admissibility might not be required in relevant practical problems and, moreover, admissibility is a sufficient condition for optimal search (with A* algorithms), but it is not necessary; there are in fact interesting cases in which non-admissible heuristics yield optimal solutions.

Many studies have been carried out on the popular Sam Loyd's sliding-tile puzzle, which turns out to be complex enough to highlight the most relevant problems. In fact, it has been proved that the attempt to solve it optimally is intractable [13]. The Manhattan Distance heuristic (MD) is of historical and theoretical importance in optimal problem solving. It had a central role, along with IDA* search algorithm, in finding the first 15-puzzle optimal solutions [7]; nevertheless no further improvements would have been achieved without investigating new heuristic sources. Manhattan's informative contribution is not sufficient to face optimally more complex problems, as Rubik's Cube or the 24-puzzle. Computational brute force is powerless in combinatorial settings, therefore the research on heuristics is of crucial importance in problem solving. Recently the field has been splitted into two guidelines each one referring to a different type of heuristic source: "online" and "memory-based" heuristics. *Online heuristics* aim to overtake MD, by automatically inventing new techniques [11], by generalizing it through a constraint addition process (Higher-Order heuristics) [9] or by enhancing its informative output with ad hoc corrections [5]. The main facet is that heuristic values are computed within search, every time a node is explored. The information bias of Manhattan's algorithm is that it considers each piece of the puzzle as independent, hence it ignores tile interactions and conflicts. The addition of corrective techniques can interestingly improve performances: Conflict Deduction [3][1] reduces IDA* search tree by nearly two order of magnitudes on the 15-puzzle. This achievement is still insufficient to challenge bigger puzzles. Moreover, hand made heuristic corrections tend to be strongly problem dependent: a more appealing approach is to define a general theory that enables automatic heuristic design. The first 24-puzzle solutions were encountered applying higher-order heuristics to IDA* [9]. The drawback of this technique is that simultaneous tile analysis with more than two tiles (i.e. Triple Distances) is NP-Complete, hence it cannot be further extended.

On the contrary, *memory-based heuristics* purpose is to map previously solved subproblems to their optimal path cost, thus using memory resources heavily. These have been successfully used to solve the Rubik's Cube [10] and the 24-puzzle [8], for which Disjoint Pattern Databases were introduced. These sharpen the preliminar technique (Pattern Databases [1]) ignoring the interactions between patterns: with this trick the distances obtained by all disjoint sets can be summed together without any risk of inadmissibility. DPDBs (with reflections) are actually the most powerful heuristic sources. This excellence is impressive for the 15-puzzle: with the same quantity of memory this technique speeded up non-additive Pattern Databases [2] by a factor of 150 (2000 times faster than simple MD) and reduced the search tree by an order of magnitude. Not the same can be said for 24-puzzle were DPDBs have displayed over higher-order heuristics a relatively small speedup (between 1,1 and 21) and larger search trees.

The complexity of most of the problems attacked in the framework of problem solving can be relaxed while looking for an approximation of the optimal solution. A nice formalization of approximate solutions is given in [12], where the notion of ϵ-admissible search is introduced. The developed theory makes it possible to establish a bound of the cost of reaching the goal. In this paper, we introduce the related concept of *likely-admissible heuristics*, where the admissibility requirement is relaxed in a probabilistic sense. Instead of providing an upper-bound to the cost, we guarantee to end up with optimal solutions with a given probability. In so doing, we provide a framework which is somehow related to PAC-learning [14]: problem solvers can be created that probably return optimal solutions in a certain portion of cases. Interestingly, likely-admissible heuristics can

[1] This assembles and generalizes five MD-corrections: linear conflicts, last moves, corner tiles, non-linear conflicts and corner deduction.

be obtained naturally by statistical learning techniques such as artificial neural networks, which can learn from examples the expected value of the cost to reach the target. In the case of Manhattan Space Problems (defined in section 2), like the sliding-tile puzzle, we prove that the strict admissibility requirement for heuristic function h can be relaxed from $h < h^*$ to $h < h^* + 2c$, where h^* is the cost of optimal path from the current state to the goal. Such a relax is profitably exploited for the learning of the heuristics, which can in fact end up with a given degree of error. We used multilayered neural networks with a proper design of the input coding and a linear output so as to perform predictions of the distance. A novel cost function is proposed, where the errors are given a different weight depending on whether or not they overestimate the solution, so as to bias the learning towards admissibility.

Our experiments with the 15-puzzle and IDA* are very promising and show that the adoption of likely-admissible sub-symbolic heuristics yield optimal solutions in 50% of the cases, taking only 1/500 time (1/13000 space) of classic Manhattan-based search.

2 Admissible overestimations in Manhattan Spaces

In the literature, there are a number of known cases of "admissible overestimations". It is straightforward, for example, that overestimations outside the optimal solution path π^* do not affect optimality [12]. Another obvious assertion is that overestimating $h^*(n)$ by a fixed constant c for all states n is influential for node expansion.

We shall call *Atomic Manhattan Space (MS) Problems* those where: (i) the problem states are defined by a set of coordinates for each atomic element of the problem (i.e. the tiles in the 15-puzzle), (ii) the operators perform only over one single element at a time and affect just one coordinate, (iii) this mono-dimensional successor function has the constant cost $g(n, \mathrm{SCS}(n)) = c \; \forall n$. Two Atomic MS Problems are Robot Navigation over a grid and the 15-puzzle.

It is easy to demonstrate that in such a context any wrong decision requires (at least) the application of an operator and its "reverse", which leads to a path with a cost $g \geq C^* + 2c$. Hence:

Theorem 1. *Let P be an* Atomic MS Problem *with transition cost c. Let h be a non admissible heuristic in P (except for the goal state, where $h(n) = 0$) with an overestimation of h^* equal to a variable s smaller than twice the operator cost ($0 \leq s < 2c$) so that $h(n) < h^* + 2c$. Heuristic h leads an A* algorithm to optimal solutions.*

Proof. The proof is given by contradiction. Let $\gamma^* \in \Gamma^*$ be the optimal goal node with a path cost C^* and $\gamma_2 \in \Gamma$ be a suboptimal goal node. Assume that n is on the optimal path π^* to γ^* and has not been expanded. Hence, since $h(n) < h^* + 2c$:

$$f(n) < C^* + 2c. \quad (1)$$

Assume that γ_2 is about to be expanded. To expand this node A* requires $f(n) \geq f(\gamma_2)$. For problem P, as stated above, $g(\gamma_2) \geq C^* + 2c$. Heuristic h is admissible for the goal state, therefore:

$$h(\gamma_2) = 0 \text{ and } f(\gamma_2) = g(\gamma_2) \geq C^* + 2c. \quad (2)$$

Equation 2 is in contradiction with 1: $f(n) \geq f(\gamma_2)$ is impossible if $f(\gamma_2) \geq C^* + 2c$ and $f(n) < C^* + 2c$. As a consequence A* will never expand γ_2 and it will follow the optimal path π^* until it reaches γ^*. □

Note that optimality might not be achieved if the overestimation equals $2c$. In such a case we could have $f(\gamma_2)$=$f(n)$=C^*+$2c$: A* could choose γ_2 for expansion and a suboptimal solution would be found. However optimality is assured even if: for each overestimation $s_1 \geq 2c$ of a node $n \in \pi^*$ there is at least one node $n' \notin \pi^*$, for each subpath of OPEN nodes (frontier), with an overestimation $s_2 > s_1$-$2c$.

Despite a strong and negative impact on search efficiency, "admissible overestimations" still force to expand all nodes belonging to the optimal solution path. We performed a simple test with A* on the 8-puzzle and noted that whilst s grows, the number of expansions increases rapidly. In allowing s=2, as expected, the search tree deflates but non-optimal solutions start to appear. With IDA* the damage is even more remarkable: if s=1 time complexity doubles because overestimations in the search frontier trigger odd intermediate iterations.

More complex problems do not undergo this theorem, but still a margin for overestimations can often be found. The Rubik's Cube, for example, is a non-Atomic MS Problem (operators move cubes in groups, not singularly): the overestimation limit here is c, which is proportionally even more promising than the 15-puzzle because of its smaller solution depth.

3 The likely-admissible search framework

This framework considers a statistical dimension of optimality: the capacity of a heuristic to lead to a certain amount of optimal solutions given some specific initial conditions. This approach differs significantly from the ϵ-admissible search [12], since the relaxation of the optimality requirement is seen in a different perspective.

An ϵ-admissible search encounters solutions that have a cost inferior to a certain bound proportional to the optimal cost, $g(\gamma) \leq (1 + \epsilon)C^*$. On the contrary a likely-admissible search guarantees to find optimal paths to the goal with a given probability p. The solutions corresponding to probability $(1-p)$ could also be bounded, but this is not strictly necessary.

We believe that the approach proposed in this paper is very promising since it yields truly optimal solutions within reasonable computational efforts and, moreover, it is quite general and can be applied to many different problems. It is also worth mentioning that the approach can take advantage of the adaptive power of general approximators like neural networks.

The likely-admissibility represents a more general framework than ϵ-admissibility. The simplest case of A^*_ϵ algorithm uses a cost function $f(n) = g(n) + wh(n)$. WA* is a generalization of A* that returns solutions with a path cost bounded by wC^*[2] only if $h(n) \leq h^*(n)$, therefore the heuristic function needs to be previously proven as admissible. On the contrary likely-admissibility can be applied to any kind of available heuristics, the only element required is some knowledge about overestimations frequency.

Let us denote with $P(h\downarrow)$ the probability that heuristics h underestimates the distance to the goal of any state $x \in X$. Only overestimations of nodes within the optimal path π^* can affect optimality. Hence the final probability p depends on $P(h\downarrow)$ and on length d of π^*. The probability p_h that an A* algorithm, guided by heuristic h, terminates any state $x \in X$ with an optimal solution is:

$$p_h = P(h\downarrow)^d \quad (3)$$

This result can be sharpened reformulating the definition of admissibility as $h(x) < h^*(x) + 2c$ (Theorem 1).

This has a great practical impact on neural heuristics (sec. 4) admissibility, as it can be seen in Fig. 1. For example the best neural net trained for the 8-puzzle overestimated h^* in 28,4% of the cases,

[2] Solution costs of WA* are much smaller in practice: ϵ-admissible upper bound is very inaccurate.

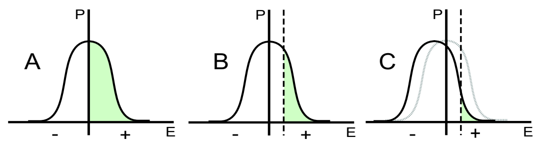

Figure 1. X-axis: heuristic estimation error; Y-axis: probability of the error. Non-admissible overestimations are highlighted in grey. **Chart A**: neural estimations are accumulated around the target with equal probability for under and overestimations. **B**: the effect of th. 1, the vertical line indicates admissible overestimations. **C**: the curve is shifted to the left because (section 4) it is possible to partially stress neural heuristics to underestimations. The grey-tagged space has been sensibly reduced.

but it overestimated $h^* + 2$ in only 1,9%. We denote this probability with $P(h{+}2\uparrow)$.

The likely-admissible framework can be enriched by using a family H of j heuristics and choosing as final estimation the lowest output. Hence, Equation 3 can be reformulated:

$$p_{H^j} = \left(1 - \prod_{h=h_1}^{h=h_j} P(h+2\uparrow)\right)^d \quad (4)$$

We shall ideally suppose that all heuristics have the same $P(h{+}2\downarrow)$: it emerges that the number of different sources needed to fulfil the p requirement grows logarithmically with [3] d and the desired p_H:

$$j \approx \lceil \log_{P(h+2\uparrow)}(1 - \sqrt[d]{p_H}) \rceil \quad (5)$$

For example: with a desired p_H=$0,999$, for the 8-puzzle 3 or 4 neural heuristics with $P(h{+}2\downarrow) \approx 0,95$ would be sufficient (for a similar accuracy n would have to grow only up to 5 for the 15-puzzle).

However, equation 4 and 5 imply a total independence between all heuristics contained in H and eq. 3 implies a constant overestimation probability, regardless of h^*. In our experiments (table 1) these assertions appeared to be optimistic. Hence, to provide some preliminary results of optimality prediction we simply used eq. 3. This turned out to be generically pessimistic[4] (a desired condition for the reliability of the framework), especially for predictions below 50%. This is quite obvious because the equation does not consider the positive effect of the overestimations outside the optimal path, therefore it can be considered as a statistical lower bound. When $P(h\uparrow)$ decreases (column n. 5, 6 and 7 of the table) the prediction becomes very precise.

4 Subsymbolic heuristics

Because of its generation process, unbind from the logical structure of the problem, neural heuristics cannot be formally admissible. Nevertheless the involvement of artificial neural networks with heuristics can be well motivated if we consider that: memory requirement grows exponentially, admissible search can tolerate a certain degree of overestimations, admissibility can be targeted through a statistical filter.

Subsymbolic heuristics can be regarded as "online heuristics", even though they require offline dataset generation and training phase. Two observations can explain our assertion. First: the ideal neural heuristics should be uncoupled from the puzzles dimensions[5] so that learning over small puzzles can be generalized without limitations. On the contrary, DB heuristics have to be generated "ex novo" for each problem. Second: during search subsymbolic heuristics face unseen configurations, whereas DB heuristics require a prior exhaustive solving of all the subgoals of the puzzle.

Another important remark is that subsymbolic heuristics represent a more general tool for heuristic generation than higher-order heuristics, because they can be applied even when a problem cannot be neatly described by its constraints or when database exhaustive preprocessing is computationally impossible.

For this work a complete system for neural heuristic generation and exploitation has been developed. Basically the model is that of a multilayer perceptron that learns from examples (optimally solved configurations), how to estimate the optimal cost from any state to the target.

Neural network-based heuristics

Hidden Units. We generated various neural nets with a single hidden layer and a number of HU varying between 3 and 20. Additional layers appeared to provide little learning power. Considering the ratio between information quality and computational cost of the heuristics, 15 emerged as the maximum number of useful units. For all HU we used the hyperbolic tangent activation function.

Outputs and targets. All networks were provided with a single output unit with linear activation. In order to avoid the risk of inner neuron saturation the linear output and the target have to be normalized, compressing the values between 0 and 1, for example using the maximum possible output value (i.e. 31 for the 8-puzzle). Two different targets have been designed. A) *"direct" target function*: $h_{ANN}(n)$=$h^*(n)$. The network has to produce the exact optimal distance between node n and the goal state. B) *"gap" target function*: $h_{ANN}(n)$=$h^*(n)$-$h_m(n)$. Here the goal is to fill up the void between h_m (a Manhattan-type admissible heuristic) and $h^*(n)$. This second target represents a first step towards the integration between symbolic and subsymbolic heuristic sources.

Inputs and coding. Given a puzzle with N squares the input coding is given by a vector of N^2 bits, where the bit $N{\times}k{+}t$ is high if the square k is occupied by the tile number t, low in any other case. This coding is very effective with the 8 and the 15-puzzle, but its polynomial growth leads to a heavy computational cost for bigger dimensions. At present we are studying, with results, a coding technique that grows nearly linearly with the number of tiles.

Learning Algorithm. The classical backpropagation algorithm was adopted along with the introduction of a novel error attribution function on the target neuron. Given an example d, an output o_d and a target t_d the error is computed as E_d=$(1{+}w)(o_d$-$t_d)$ if $(o_d$-$t_d)$>0 and E_d=$(1$-$w)(o_d$-$t_d)$ if $(o_d$-$t_d)$<0, instead of the classic E_d=$(t_d$-$o_d)$. The coefficient of asymmetry has to undergo the following constraint: $0{\leq}w{<}1$. With a w close to 1 the learning process forces the network to underestimate the target, engendering a certain slowdown of the global convergence. If w is set to 0 the basic backpropagation is restored. The asymmetric learning technique that we designed is general and can be used in any machine learning context where the direction of the error, not just its amount, matters. In our case we used a dynamic w that gently decreases during learning. To maintain the convergence smooth we used a momentum with α=$0,8$ and a dynamic learning rate η in a range between 0,1 and 1.

[3] d can be reduced shifting from a p-admissible source to a fully admissible one while approaching the solution. In all experiments carried out we shifted to Corner Deduction at $h_m \leq 5$.

[4] If $P(h{+}2\uparrow)$ is much higher for nodes that are distant from the goal than for closer nodes we could occasionally have overoptimistic predictions; mainly because if the lower part of the tree is void of overestimated frontier nodes upper non-admissible overestimations harm more.

[5] We are currently testing a recurrent ANN architecture, independent of the number of tiles because the puzzle is represented as a graph, not a vector.

Dataset generation. Two important remarks. a) Optimal learning can be achieved with a very moderate number of examples in the learning set L (around 20000 for the 15-puzzle: ≈1/500^6 of the problem space). b) L has to be representative of the configurations that are encountered during search, hence the average solution depth of the example has to be no more than half the average solution depth of the problem. The ideal example distribution of L for the 15-puzzle requires an average $d \leq 26,5$ and more than 60% of examples with $d \leq 30$ (only 0,1% of the state space has a d below this cutoff).

"A posteriori" optimality handling. Various techniques have been tested to improve the admissibility of the heuristic evaluation given by the network. Best performances have been given by: a) truncating the estimation (IDA* is inefficient with a non-discrete $f(n)$); b) adapting the output to the parity of Manhattan values, MD always underestimates h^* by an even quantity (this technique can halve the number of IDA* iterations); c) terminating the search with the use of a symbolic heuristic (i.e. MD is perfectly informed when $h(n)<5$).

Parallel neural heuristics. Neural networks that during learning differ only in weight initializations lead to rather related heuristics. A way of increasing independence is the use of multiple learning sets. With these L-independent neural heuristics optimality gets closer to equation 4. The price to pay is a major cost of training set generation. The heuristic computational cost grows linearly with network additions, with parallel computing this growth can be set to zero.

5 Experimental Results

Environment description

Network training and puzzle-solving tests were implemented on a 500MH PC with a Java application (256MB RAM occupation). The speed of the machine can be observed by means of the number of nodes per sec expanded in an IDA* search with MD: over 56000 for the 15-puzzle; by comparison, Korf's C application on Sun Ultra 10 run 135 times faster (over 7 million nodes/sec.). Neural heuristics are more complex to compute (i.e. the forward phase of a 15-HU-net is 15 times slower than MD). Nevertheless the system was able to retrieve 15-puzzle optimal solutions in few seconds. Conflict Deduction was used to solve examples for training and test sets.

8-puzzle

In this preliminary test field neural heuristics have performed with astonishing accuracy. Two learning sets were generated, L_1 (12000 examples) and L_2 (9500), a validation set V (2500), a test set T (2000) with average solution depth 21,97. We trained four nets on L_1 and two on L_2. Three adopted the "gap" target function.

Learning evaluation. Non-admissible overestimations over V were contained between $1,9\%$ and $5,2\%$, the average heuristic error varied between $1,3$ and $1,7$.

Test evaluation. A* performed better than IDA* because of the strong search cutback. The best single-net heuristic reduced Manhattan's tree by a factor of 12 (just above 100 nodes) and time by $4,5$, giving solutions with average length $22,27$: 15% were non-optimal but only $0,005\%$ needed more than 2 additional moves.

Parallel nets strongly improved accuracy. The combination of two nets of the same L more than halved the number of overestimations on V. With two L-independent nets the number of optimal solutions with A* was already over 94% and the size of the search tree surprisingly decreased. The contemporary use of all six neural heuristics lead to a 98% optimality degree (avg. $d = 22,014$) whereas the number of nodes generated grew gently, 133, still less than some of the single heuristics, which is positively surprising. The choice of the lowest value among nets estimations increases the average heuristic error but this can, paradoxically, reduce search complexity. It is not a contradiction: it indicates that a more efficient heuristic function has been generated, as illustrated in [6]. This remarkable phenomenon has occurred with the 15-puzzle too.

The capacity of equation 3 to predict factual optimality is very interesting for the 8-puzzle: no optimistic forecasts were produced and for estimations greater than 80% the gap never exceeded 4%.

15-puzzle

Learning evaluation. The learning environment comprehended two training sets L_1 and L_2 with 25000 cases. V (5000) was generated with the same example distribution of L_1 and L_2, as specified in section 4. We trained two nets on L_1 and two on L_2 (with a number of hidden units between 5 and 15). The dataset generation process took around 100 ours whereas the learning phase about 200. However it has to be noted that this time requirement can be linearly reduced with parallel computation. Despite the small number of examples the training generated highly informed heuristics. The average estimation error over V was settled between 2,3 and 2,6: the heuristics captured over 91% of the desired information (avg. 27 moves). This is very encouraging because it improves the results obtained by learning over the 8-puzzle domain, where the information gain was 86%. The percentage of non-admissible overestimations was contained between 4% and 6%, a gentle growth with respect to the 8-puzzle. Learning and test evaluations can both be observed in table 1.

Test evaluation. We will present here plain IDA* results for a clearer literature comparison. The test set T (700 examples) was generated randomly and produced an average d of 52,62.

As it can be noticed in the table, all the four nets, singularly terminated with optimal solutions in nearly one third of cases. These are the first non database heuristic solutions that can challenge in performance Disjoint Pattern DB results. The search tree contains between 24 and 33 thousand nodes. Neural heuristic A reduced Manhattan's space cost by a factor of 15000 and by three order of magnitude its running time (better than DPDB, but slightly slower than DPDB with reflection). These results are even more impressive if we consider that admissible-overestimations (around 10% of cases with the output truncament) have a great negative impact over search efficiency.

By combining two L-independent neural heuristics we optimally solved around 50% of configurations (95% within C^*+2). The optimality degree appeared to be nearly uniform among different solution depths. This 0,5-admissible search was performed with non-growing space complexity (still 20% better than DPDB with reflection). Finally, we wanted to stress the solution optimality. The full set of neural heuristics produced a quantity of optimal solutions close to two thirds, with a 33% growth of node expansion. Further, an arbitrary 1-move-reduction was added to h_{ANN}. This tripled time and space costs, but the optimality degree approached 90%.

Optimality predictions (third row of the table) have proved to be rather accurate for parallel neural heuristics (the four columns on the right) and in only one case the prediction was optimistic (heuristic ABCD). To clarify this rare occurrence we are currently studying how the heuristic error affects the effective optimality of search. On the contrary the ϵ-admissible framework provides a very inaccurate prediction theory. For example WA* with MD and W=2 solves the 15-puzzle with an average d of 63,5 [4], which is by far lower than the predicted: 53×2=106. Plain A* with a single neural heuristic is remarkably more efficient than WA* with Manhattan. In our tests A* expanded 12492 nodes with d=54,7, whereas, WA* needs to expand

Neural Heuristics $\Rightarrow$	A(L_1,15hu)	B(L_1,5hu)	C(L_2,12hu)	D(L_2,15hu)	AB	AC	ABCD	ABCD-1
V, Avg. Estimation Error	2,32	2,58	2,37	2,34	2,77	2,81	3,24	4,14
V, Overestimations ($P(h$+2$\uparrow$))	5,34%	4,15%	5,30%	5,77%	2,20%	1,65%	0,69%	0,26%
V, Optimality prediction (eq.3)	5,5%	10,6%	5,6%	4,3%	32,1%	41,4%	69,2%	86,9%
T, Factual Optimality	28,71%	29,14%	28,43%	26,0%	40,57%	48,86%	63,86%	89,14%
T, Avg. Solution Length (d)	54,45	54,38	54,47	54,52	54,00	53,75	53,40	52,83
T, Avg. Nodes Visited	24711	33874	29106	28489	34290	29189	43094	132895
T, Avg. Time (seconds)	7,38	5,36	7,45	8,51	16,81	14,32	36,47	111,39
T, ANN/MD: space cost ratio	1/15000	1/11000	1/12700	1/13000	1/11200	1/12700	1/8500	1/2800
T, ANN/MD: time cost ratio	1/1000	1/1400	1/1000	1/525	1/450	1/520	1/200	1/70

Table 1. Single neural heuristics are named by letters A, B, C and D. Their training set and their number of hidden units are between brackets. Heuristic AB indicates the parallel neural heuristic obtained joining A and B. ABCD-1 indicates the introduction of an arbitrary 1-move-reduction on the final estimation. The average error and the number of non-admissible overestimations (that produce he optimality prediction in row 3) were calculated over the validation set V. All the others results were obtained over the test set with IDA*. Comparisons with Manhattan are approximated because T was solved with Conflict Deduction.

nearly 500 thousand nodes to preserve d below 57 (W_g=2, W_h=3). This gap is even more evident if we consider WIDA* performances.

6 Conclusions

We presented a novel approach to problem solving in a statistical perspective by introducing the concept of likely-admissible heuristics, which allow one to discover optimal solutions in a fraction of the problem instances, while saving significant computational resources with respect to the A* algorithm equipped with strictly admissible heuristics. The proposed general framework is very well suited for subsymbolic heuristics that can be generated using machine learning models, such as artificial neural networks, over a dataset of previously solved examples. We report experimental results for the Sam Loyd puzzles (8 and 15 tiles). The results are very interesting and can be read from two different perspectives.

1. Analysis of the informative power of neural heuristics.
   The experiments stated that neural heuristics are the first online sources capable to return a certain amount (29%) of optimal solutions of the 15-puzzle with time and space costs lower than Disjoint Pattern Databases (non-reflected). When coupling two neural networks for the computation of the heuristics, the optimality degree reached approximated 50% with non growing search complexity. Compared to Manhattan Distance this search performed over 500 times faster by using 1/13000 memory resources.
2. The competence of our predictive theory.
   This gave a precise lower-bound of the effective optimality in 8-puzzle solving. In the 15-puzzle context, an unwished optimistic estimation has been observed, which means that the predictive tools have to be sharpened more. Nevertheless the framework is more precise than ϵ-admissibility, which greatly overestimates the final solutions costs.

7 Further work

We have previously mentioned that *generalization of the input*: the neural network input coding should be uncoupled from puzzle's dimensionality in order to exploit the learning performed on smaller puzzles. Another main goal is to face the 24-puzzle, but this requires the prior elimination of three obstacles: a) Currently neural heuristics depend on external optimal sources for dataset generation. This obstacle can be removed. A possible solution is *auto-feed Learning*: first, networks are trained on small sets of simple configurations; then they automatically generate their own training sets, solving autonomously cases with increasing depth solutions. Suboptimal examples would be rare because of the strong underestimation tendency that derives from previous training cycles over easier cases. b) Neural heuristic overestimations are not independent, this impedes the use of equation 4. c) The admissibility of h dependends on the solution cost. This defect could be softened by partitioning the training set in small subsets and performing a *network specialization* training over a certain class of configurations. This would produce less overestimations, requiring considerably less examples and learning time. This technique also enables to analyze how h^* affects the admissibility of the heuristic, hence it should help to sharpen the optimality prediction.

Likely-admissibility depends on the solution depth of the problem and not on the branching factor. For this reason, likely-admissible neural heuristics should display their best results over puzzles with big branching factors, like Rubik's Cube.

References

[1] J. C. Culberson and J. Schaeffer, 'Searching with pattern databases', in *Canadian Conference on AI*, pp. 402–416, (1996).

[2] J. C. Culberson and J. Schaeffer, 'Pattern databases', *Computational Intelligence*, **14**(3), 318–334, (1998).

[3] M. Ernandes, 'Joining symbolic and subsymbolic sources for p-admissible non memory-based heuristics.', *AI*IA Notizie*, **4**, (2003).

[4] Ariel Felner, S. Kraus, and R. E. Korf, 'Kbfs: K-best-first search', *Annals of Mathematics and Artificial Intelligence*, **39**(1-2), 19–39, (2003).

[5] O. Hansson, A. Mayer, and M. Yung, 'Criticizing solutions to relaxed models yields powerful admissible heuristics', *Information Sciences*, **63**(3), 165–178, (1992).

[6] R. C. Holte and I. T. Hernádvölgyi, 'A space-time tradeoff for memory-based heuristics', in *Proceedings of AAAI-99*, pp. 704–709, (1999).

[7] R. E. Korf, 'Depth first iterative deepening: An optimal admissible search.', *Artificial Intelligence*, **27**, 97–109, (1985).

[8] R. E. Korf and A. Felner, 'Disjoint pattern database heuristics', *Artificial Intelligence*, **134**, 9–22, (2002).

[9] R. E. Korf and L. A. Taylor, 'Finding optimal solutions to the twenty-four puzzle', in *Proceedings of AAAI-96*, pp. 1202–1207, (1996).

[10] Richard E. Korf, 'Finding optimal solutions to rubik's cube using pattern databases.', in *Proceedings of AAAI-97*, pp. 700–705, (1997).

[11] J. Mostow and A. Prieditis, 'Discovering admissible heuristics by abstracting and optimizing: a transformational approach', in *Proceedings of IJCAI-89*, pp. 701–707, Detroit, (1989).

[12] J. Pearl, *Heuristics*, Addison-Wesley, Reading, MA, 1984.

[13] D. Ratner and M. Warmuth, 'The (n^2-1)-puzzle and related relocation problems', *Journal of Symbolic Computation*, **10**, 111–137, (1990).

[14] L. Valiant, 'A theory of the learnable.', *Communications of the ACM*, **27**(11), 1134–1142, (1984).