Application of Causal Models for the Selection and Redesign of Heuristic Algorithms for Solving the Bin-Packing Problem

Joaquín Pérez[1], Laura Cruz[2], Rodolfo Pazos[1], Vanesa Landero[1], Verónica Pérez[2]

Gerardo Reyes[1], Jorge Ruiz Vanoye[1]

[1] Departamento de Ciencias Computacionales, Centro Nacional de Investigación y Desarrollo Tecnológico (CENIDET), AP 5-164, Cuernavaca, México

[2] División de Estudios de Posgrado e Investigación y Departamento de Sistemas y Computación, Instituto Tecnológico de Ciudad Madero (ITCM), Cd. Madero, México

Abstract

This paper deals with the problem of selecting heuristic algorithms for solving NP-hard problems using a causal approach. The first works pursued finding an algorithm which was the best for solving some problem. Subsequent works showed the superiority of an algorithm applied only to some instances subset. However, those works have been very limited in explaining why an algorithm outperforms another. In this paper a causal approach is proposed for providing explanations. It was applied to two variants of the Threshold Accepting algorithm and two variants of the Tabu Search algorithm, in solving the Bin Packing problem. As a result, a causal model was generated for each algorithm, whose interpretation showed quantitatively the factors that contributed the most to the superiority for each algorithm. Finally, we claim that this approach can be useful for the selection and redesign of algorithms.

Keywords: causal modeling, learning techniques, algorithm selection, behavior explanation.

Introduction

A good alternative for solving very large instances of combinatorial optimization problems are heuristic algorithms[1]. Unfortunately, in real life situations, there is usually no algorithm that outperforms all the other algorithms for all instances[2], and therefore, the problem of selecting the best algorithm arises. Several related works (Table 1) have experimentally analyzed algorithm behavior in order to find the best algorithm. Columns 2 and 3 indicate whether the algorithm analysis includes information from the parameter description (IPD) or from a sample of the solution space (IPS) of a problem instance. Columns 4, 5 and 6 indicate if the algorithm behavior (IAB), search trajectory (IST) and algorithm structure (IAS) are considered in the analysis. Column 7 indicates if the works present formal explanations. A survey of the specialized literature revealed the inexistence of a formal model that explains the association between indicators of problem instances and indicators of an algorithm that

---

[1] Garey M. R.: *Computers and Intractability, a Guide to the Theory of NP-completeness*, W. H. Freeman and Company, 1979.

[2] Wolpert D. H., Macready W.G.: *No Free Lunch Theorems for Optimization*, IEEE Transactions on Evolutionary Computation, vol. 1, pp 67-82, 1997.

solves the instances successfully. Therefore, the problem of explaining why an algorithm dominates in an instance region is approached in this paper. The solution to this problem is important, since it may provide a solid foundation for the selection and redesign of algorithms for solving given instances of NP-hard problems. In this sense, the solution approach presented permits systematically finding relations between influencing indicators (columns 3-6) of dominance of an algorithm and the inner workings of algorithms (column 7), in order to provide formal explanations through causal analysis.

Table 1. Related works

Work	Problem Instances Indicators		Algorithm Indicators		IAS	Formal Explanation
	IPD	IPS	IAB	IST		
Soares[3]	✓					
Pérez[4]	✓					
Hoos[5]	✓			✓		
Lemeire[6]			✓			✓
Pérez[7]		✓	✓			✓
This paper	✓	✓	✓	✓	✓	✓

Problem Description

The problem of explaining formally why an algorithm outperforms others in solving an instances subset can be described formally as follows:

Let:

$I = \{i_1, i_2, ..., i_m\}$ a set of instances of problem P,

$R = \{R_1, R_2, ..., R_n\}$ a partition of I,

$A = \{a_1, a_2, ..., a_n\}$ a set of algorithms,

$fp(a_q, i)$ a function that evaluates the performance of an algorithm $a_q \in A$ when solving an instance $i \in I$, then,

$$B = \{(a_q \in A,\ R_q \in R) \mid\ fp(a_q, i) > fp(\alpha, i)\ \ \forall\ \alpha \in (\mathrm{A} - a_q), \forall\ i \in R_q\ \} \tag{1}$$

i.e., B is a set of ordered pairs (a_q, R_q), where each dominant algorithm $a_q \in A$ is associated to one element R_q of partition R. In the following sections each ordered pair (a_q, R_q) will be used to mean the domination (superiority) region R_q of algorithm a_q, and set B will stand for the domination regions of the algorithms.

The problem can be stated as follows: is it possible to find relations between indicators that characterize instances I, and indicators that characterize the behavior of an algorithm a_q

[3] Soares C.: *Ranking Learning Algorithms: Using IBL and Meta-Learning on Accuracy and Time Results*, Journal of Machine Learning vol. 50, No 3, pp 251-277, 2003.

[4] Pérez O., Pazos, R.: *A Statistical Approach for Algorithm Selection*, Lecture Notes in Computer Science, vol. 3059, pp 417-431, 2004.

[5] Hoos H., Smyth K.: *Search Space Features Underlying the Performance of Stochastic Local Search Algorithms for MAX-SAT*, 2004.

[6] Lemeire J., Dirkx E.: *Causal Models for Parallel Performance Analysis*, 2004.

[7] Pérez J.: *Explaining Performance of the Threshold Accepting Algorithm for the Bin Packing Problem: A Causal Approach*, 14th International Multi-conference, Advanced Computer Systems, Polish Journal of Environmental Studies, 2007.

through a formal model M? Will model M provide solid foundations to explain why algorithm a_q is the best for solving instances in subset R_q and not instances in subset $(R_q)^c$?

Causal Models

Definition. A causal model can be defined as a causal Bayesian network[8] $M = (V, G, F)$,

- $V = \{v_1, v_2, \ldots, v_n\}$ is a set of observed discrete variables.

- G is a directed acyclic graph with nodes corresponding to the elements of V that represents a causal structure $C = (V, E)$; i.e.,

$E = \{E_1, E_2, \ldots, E_n\}$, where each $E_i \in E$, $E_i = \{(v_i, y_1), (v_i, y_2), \ldots, (v_i, y_n)\}$ is a set of ordered pairs, where (v_i, y_k) is in E_i if and only if y_k is a direct cause of v_i relative to V, and there is a directed edge from y_k to v_i in G, where $v_i \in V$ and $v_i \neq y_k$. Also, $Pa(v_i) = \{y_1, y_2, \ldots, y_n\}$ is a set of direct causes of v_i.

- $F = P(v_i \in r_{vi} \mid y_1 \in r_{y1}, y_2 \in r_{y2}, ..., y_n \in r_{yn})$ is a function of conditional probability of variable v_i in the value range r_{vi} given the direct causes (parents) of v_i in the value ranges $r_{y1}, r_{y2}, ..., r_{yn}$; where v_i has u ranges of values, and each direct cause $y_k \in Pa(v_i)$ has at least w ranges.

The edges in graph G are interpreted causally if the causal model M guarantees the causal Markov, minimality and faithfulness conditions. $P(v_i \mid Pa(v_i))$ represents a stochastic process by which the values of v_i are chosen in response to the values of $Pa(v_i)$. This process must stay invariant under interventions in processes governing other variables. The joint probability distribution of the domain variables must be factorized as expression (2)[8].

$$P(V) = \prod_i P\left(v_i \mid Pa\left(v_i\right)\right) \tag{2}$$

Construction of Causal Models.

The process of generating causal models is a NP-hard problem[9]. Causal modeling generally has four stages[10]. First (*specification*): the variables that are causes and effects are determined; to this end, a structure learning algorithm is used to discover a graph G that represents a causal structure $C = (V, E)$. Second (*estimation*): a parameter learning algorithm applies F and calculates the intensity of the causal relations found. Third (*interpretation*): the most important relations with the highest magnitude are analyzed, and the final explanations are given by the researcher in the context of previous knowledge of the problem domain. Fourth (*evaluation*): the model is tested to assess its prediction accuracy.

Analyzing the Performance of the Threshold Accepting and Tabu Search Algorithms

The general solution approach proposed, shown in Fig. 1, is described and validated through

---

[8] Spirtes P., Glymour C.: *Causation, Prediction, and Search*. MIT Press, 2nd Edition, 2001.

[9] Chickering D.: *A Transformational Characterization of Equivalent Bayesian Network Structures*, 1995.

[10] Cohen P.: *Empirical Methods for Artificial Intelligence*, The MIT Press Cambridge, 1995.

two study cases. The first involves two variants (a_1, a_2) of the Threshold Accepting algorithm (TA)[4], and the second two variants (b_1, b_2) of the Tabu Search algorithm (TS)[4] (in this case, only the results are mentioned). Both algorithms are applied to the solution of 324 instances of the Bin Packing problem, which were selected randomly from OR library[11,12].

1. Description and space of the problem. Indicator *oac* represents the problem description (*IPD*) and characterizes the parameters of each problem instance; it consists of the proportion of the total size of the objects that can be assigned to one container[4]. A sample of the solutions space (*IPS*) of each problem instance is obtained prior to the algorithms experimentation. It is built by generating 100 random solutions, where each solution x is evaluated by the fitness function $f(x)$[13]. The variability *psv* of these values is calculated.

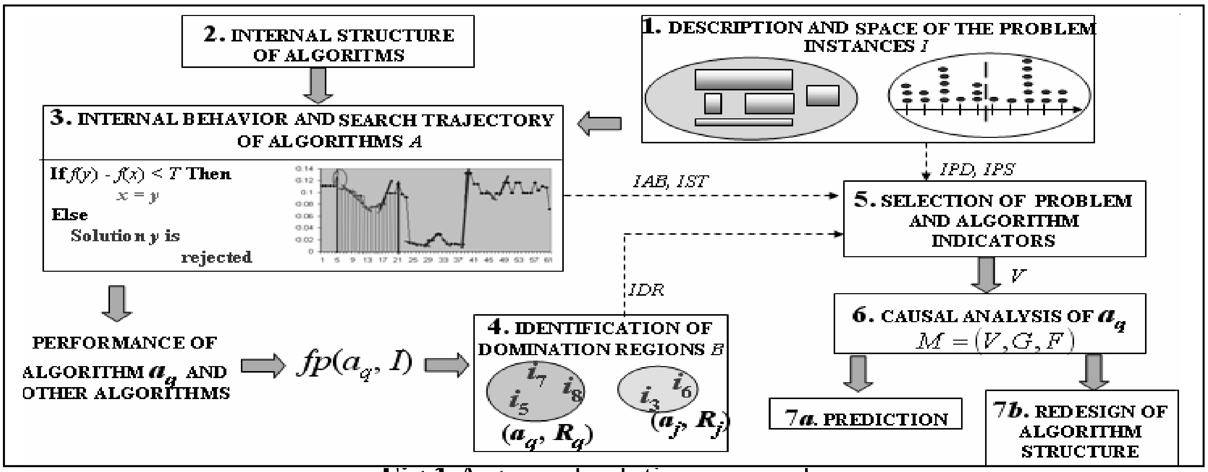

Fig 1 A general solution approach.

2. Internal structure of algorithms. The internal structure of two variants is distinguished by the method for generating neighbor solutions. Variant a_1 uses only one method: swap (1, 0)[14]. Variant a_2 uses several alternative methods (swap (1, 0), swap (1, 1), swap (1, 2), swap (2, 2), swap (0, 1), and swap (2, 1)); it tries initially the first method, if it can not generate a neighbor solution; then it tries the second one, and so on.

3. Internal behavior and search trajectory of algorithms. Algorithms a_1 and a_2 were executed 15 times (in the pilot study we observed a very small variance of these runs) for each problem instance. The information of variants a_1 and a_2 was characterized. The number of solutions accepted by the algorithm and the number of generated temperatures are recorded by indicators *asn* and *gtn* (*IAB*). The trajectory traced (*IST*) is characterized in two ways: 1) two known indicators were used, coefficient *ac* and the length *al* of autocorrelation[5]; 2) three indicators were proposed, number of inflexion points *pn*, number and size of valleys *vn*, *vs*.

---

11 Beasley J.: *OR-Library*. http://people.brunel.ac.uk/~mastjjb/jeb/orlib/binpackinfo.html.
12 Scholl A., Klein R.: http://www.wiwi.uni-jena.de/Entscheidung/binpp/.
13 Falkenauer E.: *A Genetic Algorithm for Bin Packing and Line Balancing*, IEEE Computer Society Press, pp 1186-1192, 1992.
14 Fleszar K., Hindi K. S.: *New Heuristics for One-dimensional Bin Packing. Computers and Operations Research*, 2002.

4. Identification of domination regions. Once the solution process ends, the solution quality and execution time are calculated for each algorithm. The first (*quality*) is the ratio of the best solution found by the algorithm to the theoretical solution; this solution is the summation of the object sizes divided by the containers capacity. The second (*time*) is the number of evaluations of the fitness function for feasible and infeasible solutions. A function $fp(a_q, i)$ is used to evaluate the performance of each variant a_q for each problem instance i in terms of *quality* and *time*. Expression (3) defines this function. The dominance region R_q of variant a_q is the set of all instances $i \in I$, where algorithm a_q was the best for solving them (4). The domination success sR_q of algorithm a_q is the percentage of instances that lie in region R_q. For algorithms a_1 and a_2 we had: sR_1=54.6% and sR_2=45.4%. Therefore, in this study case we analyze variant a_2 to explain why it is superior with respect to a_1.

$$fp(a_q, i) = \begin{cases} 1 & \text{if algorithm } a_q \text{ has the best } \textit{quality} \text{ among all the algorithms for instance } i \\ 1 & \text{if algorithm } a_q \text{ has the smallest } \textit{time} \text{ among all the algorithms (when they have the same } \textit{quality}\text{)} \\ 0 & \text{otherwise} \end{cases} \tag{3}$$

$$R_q = \{ i \in I \mid fp(a_q, i) > fp(b, i) \;\; \forall \; b \in (A - a_q) \} \tag{4}$$

5. Selection of problem and algorithm indicators. Discretization was carried out to established several levels for the indicators. These were analyzed to identify those that had some effect on the algorithm performance (*quality*). We performed graphic and statistical analyses. An example of these analyses can be found in our previous work[7]. The problem and algorithm indicators that turned out to be relevant were *oac*, *psv*, and *asn*, *gtn*, *pn*, *vs*, *ac*, *al*.

6. Causal Analysis.

Specification of causal order. The two kinds of indicators about the trajectory traced (*IST*) by the algorithm (*ac, al* and *pn*, *vs*) yield the same information. Therefore, we generated two different causal models: the first utilized *oac*, *psv*, *asn*, *gtn*, *ac* and *al*; the second used *oac*, *psv*, *asn*, *gtn*, *pn* and *vs*. The construction of causal models was carried out using the causal inference software TETRAD (www.phil.cmu.edu/projects/tetrad_download/) and the structure learning algorithm PC[8] with a confidence level of 95%. The PC algorithm starts by constructing a complete undirected graph, then thins that graph by removing edges with zero order conditional independence relations, thins again with first order, and so on. Afterwards, it orients the edges with the evidence found. The remaining undirected edges are oriented making sure that no directed cycles occur. Finally, a graph G is discovered which represents a causal structure $C = (V, E)$. The models generated were confirmed using the causal inference software HUGIN (Hugin Expert www.hugin.com). Figures 2 and 3 show the first and second causal models. It can be observed in Figure 2 that the first causal model did not provide relevant information about direct causes of the algorithm performance (*Pa*(*fp*)), in terms of the algorithm search trajectory (*ac, al*). In contrast, the second causal model (Figure 3) shows that indicators *oac*, *gtn*, *pn* and *vs* are direct causes (*Pa*(*fp*)) of superiority (*fp*=1) or inferiority (*fp*=0) of variant a_2. Therefore, this model is considered for the next analyses.

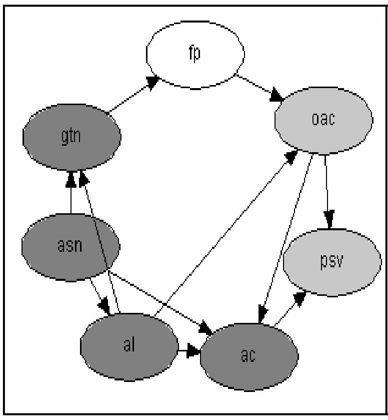

Fig 2 First causal model

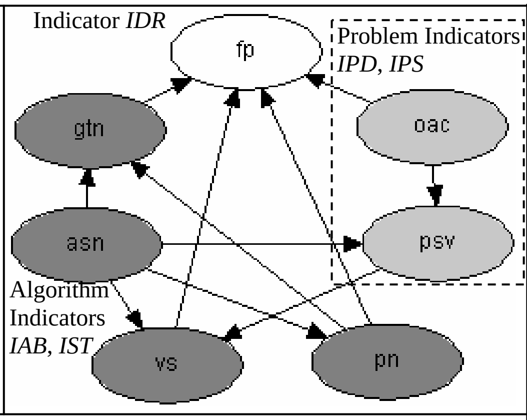

Fig 3 Second causal model

Indicators
oac. Proportion of objects that can be assigned to one container.
psv. Variability of the problem space solutions
asn. Number of accepted solutions.
gtn. Number of temperatures generated
pn. Number of inflexion points
vs. Average size of valleys
al. Autocorrelation length
ac. Autocorrelation coefficient

Model Estimation. Tables of conditional probability (CPT) of the indicators were calculated using the parameter-learning algorithm Counting[15], which applies function *F* to all the causal relations in the model. We focus on the most important magnitudes of the direct causes of node *fp*. These and their experience (*Exp*) are shown in Table 2.

Table 2. Causal relations

	Causal Relations	% Probability	*Exp*
1	P(*region*=1 |*oac*=2,*gtn*=3, *pn*=3, *vs*=2)	100	32
2	P(*region*=1 |*oac*=2, *gtn* =2, *pn*=2, *vs*=2)	75.47	53
3	P(*region*=0 |*oac*=1, *gtn* =1, *pn*=1, *vs*=1)	86.66	45

Model Interpretation. Algorithm a_2 wins for problems where the overall sum of object sizes is much larger than the containers capacity, so there is a large variety of possible distributions of objects into the containers. This algorithm is superior in solution quality. Its structure permits to intensify the search space (larger number of temperatures) and the trajectory traced by the algorithm (many inflexion points and large valley sizes) is better suited to the problem space (Relations 1 and 2). The algorithm loses (timewise) with respect to variant a_1 for problems where the overall sum of object sizes is close to the containers capacity, so there exists little variety of possible distributions of objects (few inflexion points and small valley sizes), and there is no need to intensify the search (small number of temperatures). However, its exhaustive attempts to generate neighbor solutions incur additional processing time.

7*a*. Prediction. We used the NETICA software (Norsys Corporation, www.norsys.com) to

---

[15] Korb Kevin, Ann E. Nicholson.: *Bayesian Artificial Intelligence*, Chapman and Hall, London, UK, 2004.

test the model generated on some instances, for which we ignored whether variant a_2 would be superior or not. We obtained a prediction percentage of 79.01%, which surpasses the value obtained by the Naïve Bayes classifier (77.46%). This result could justify the use of an algorithm to solve an instances subset and contribute to the algorithm selection problem.

7*b*. Algorithm Structure Redesign

The Threshold Accepting algorithm is described hereupon. Variant a_2 uses several methods for generating neighboring solutions (lines 6-7). The conclusions of the previous section allow us to redesign a_2 (ra_2) for improving its performance (lines 8-11 instead of lines 6-7).

```
1   Begin
2     T=initial temperature; μ=0.85(freezing factor);S =100(neighborhood size); x=x*(random initial solution)
3     Repeat
4       Repeat
5         For i = 1 to S
6           Build neighbor solution y of x using  swap (1, 0) or swap (1, 1)                  } a2
7                 or swap (1, 2) or swap (2, 2) or swap (0, 1) or swap (2,1)                   }
8           Calculate oac from parameters of problem instance                                  }
9           If (oac=1) Then Build neighbor solution y of x using  swap (1, 0) or swap (1, 1)   } ra2
10                          or swap (1, 2) or swap (2, 2) or swap (0, 1) or swap (2,1)         }
11                    Else Build neighbor solution y of x using swap (1, 0)                    }
12          If (f(y) – f(x)) < T then  x = y
13          Else the solution y is rejected
14      Until thermal equilibrium is reached
15       T = μT
16    Until freezing is reached
17  End
```

Preliminary experimental results show that the new variant ra_2 yields a performance improvement with respect to variants a_1 and a_2. The domination success of variant ra_2 is sR_3=68.82% (223 out of 324) when contending against variant a_1 and sR_3=59.88% (194 out of 324) when contending against variant a_2. The new redesign proposal ra_2 yields a performance improvement of 38% and 20% with respect to variants a_1 and a_2.

Other results. The proposed approach was also applied to two variants (b_1, b_2) of the Tabu Search algorithm, which are distinguished by the method for generating neighbor solutions (similar to the variants of the TA algorithm). We analyzed variant b_2, generated a causal model and proposed a new redesign rb_2. We obtained a prediction percentage 78.04%, which surpasses the value 75.92% (Naïve Bayes). The proposed redesign yields a performance improvement of 77% and 18% with respect to variants b_1 and b_2. These remarkable results show the viability of applying the proposed approach for at least two different heuristic algorithms, so as to justify their use on an instance subset, which constitutes an important contribution to the algorithm selection problem.

Conclusions

This work presents a new approach for solving the problem of explaining why an algorithm outperforms another on a set of instances using causal analysis, which yielded encouraging results. One of the main contributions of this work is the development of indicators that characterize problem instances and indicators of the behavior and performance of the algorithm. For validating the proposed approach, a set of experiments were carried out for generating a causal model that shows the interrelation of the proposed indicators, permitting to obtain formal explanations about the behavior and performance of the Threshold Accepting and the Tabu Search algorithms. We obtained prediction percentages of 79.01% and 78.04% using the models generated for these algorithms. The formal explanations found permitted to devise an improvement to the logic of these algorithms with a 20% and 18% increase in domination success. Therefore, the proposed approach permits: a) redesign proposals for the internal logic of algorithms for improving their performance, and b) understanding and formalizing the general problem of heuristic algorithm selection for NP-hard problems.

References

1. Garey M. R., Jhonson D. S.: *Computers and Intractability, a Guide to the Theory of NP-completeness*. W. H. Freeman and Company, New York, 1979.
2. Wolpert D. H., Macready W.G.: *No Free Lunch Theorems for Optimization*. IEEE Transactions on Evolutionary Computation, vol. 1, pp 67-82, 1997.
3. Soares C.: *Ranking Learning Algorithms: Using IBL and Meta-Learning on Accuracy and Time Results*. Journal of Machine Learning vol. 50, No 3, pp 251-277, 2003.
4. Pérez O., Pazos, R.: *A Statistical Approach for Algorithm Selection. Third Workshop on Efficient and Experimental Algorithms*. Lecture Notes in Computer Science, Springer-Verlag, vol. 3059, pp 417-431, 2004.
5. Hoos H., Smyth K., Stutzle T.: *Search Space Features Underlying the Performance of Stochastic Local Search Algorithms for MAX-SAT*. Parallel Problem Solving from Nature VIII. Lecture Notes in Computer Science, vol. 3242, pp 51-60, 2004.
6. Lemeire J., Dirkx E.: *Causal Models for Parallel Performance Analysis*. 4th PA3CT Symposium, Edegem, Belgica, 2004.
7. Pérez J., Cruz L., Landero V.: *Explaining Performance of the Threshold Accepting Algorithm for the Bin Packing Problem: A Causal Approach*. 14th International Multi-conference, Advanced Computer Systems, Polish Journal of Environmental Studies, 2007.
8. Spirtes P., Glymour C.: *Causation, Prediction, and Search*. MIT Press, 2nd Edition, 2001.
9. Chickering D.: *A Transformational Characterization of Equivalent Bayesian Network Structures*, 11th Conference on Uncertainly AI. San Francisco, pp. 87-98, 1995.
10. Cohen P.: *Empirical Methods for Artificial Intelligence*, The MIT Press Cambridge, Massachusetts, London England, 1995.
11. Beasley J.: *OR-Library*. http://people.brunel.ac.uk/~mastjjb/jeb/orlib/binpackinfo.html.
12. Scholl A., Klein R.: http://www.wiwi.uni-jena.de/Entscheidung/binpp/.
13. Falkenauer E., Delchambre A.: *A Genetic Algorithm for Bin Packing and Line Balancing*. IEEE 1992 International Conference on Robotics and Automation, IEEE Computer Society Press, pp 1186-1192, 1992.
14. Fleszar K., Hindi K. S.: *New Heuristics for One-dimensional Bin Packing. Computers and Operations Research*, Elsevier Science Ltd, vol. 29, pp 821–839, 2002.
15. Korb Kevin, Ann E. Nicholson.: *Bayesian Artificial Intelligence*, Chapman and Hall, London, UK, 2004.