Giochi matematici e problem solving

Project Work of Artificial Intelligence

Alejandro Sánchez Marcos
José Pizzano Martín

Index

Abstract

In this work we focus on studying the different types of games that are exposed in mathematical competitions celebrated by Bocconi University, and presenting different ways of solving most of them, and finding the limitations and advantages of each resolution method
Keywords: Logic, Problem-Solving, Constraint Satisfaction Problem, Backtracking, Heuristics, Linear Programming

Introduction

Games take part on the nature of the human been, we use it to learn and develope knowledge of our reality.
That is why we actually keep applying even learning for competitions where we check our capacities of logic reasoning.
In this case, we will use as base of our work a collection of problems that the University of Bocconi propose every year in different competitions, both in Italy and international countries.
Those problems are classified by difficulty differentiating them for levels of study.
Mathematics and games are closely linked, so much that even on the mathematics exists a branch dedicated to it . That is why we are proposed to know if a software of general resolution could be done for this mathematichal problems.

Competitions and types of problems

University of Bocconi makes various types of competitions. All of them take part between similar ages which compromise the latest stage of primary education and first ages of secundary education.
Insiede the univesity page, those competitions have different examples of the games that can we done, such as:

- Campionati internazionali
- Giochi d'autunno
- Gara a squadra
- Giochi di Rosi.

The objective of this work does not focus on the analysis of the particpants and the difficult of the games, but in the problems, facilities or difficulties of extracting information based on statements and logic resolution.

Data extraction study

On this work we will analize three competitions: Semifinals 2019, Semifinals 2018 and Giochi d'autunno 2010 (all presented on PDF format).
First of all, we are going to study the data structure on different competitions both in images as verbatim.

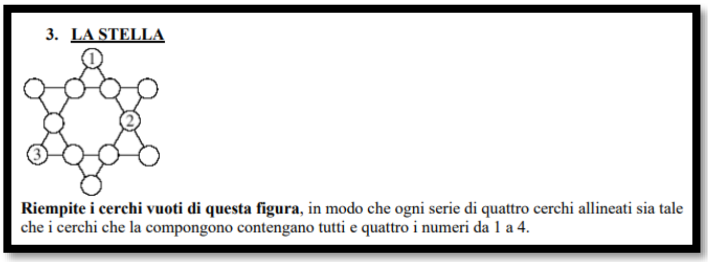
Semifinals 2019	Semifinals 2018	Giochi d'autunno 2019
18 problems	17 problems	18 problems
11 problems with image • 2 graph images • 8 geometric images • 1 matrix image	9 problems with image • 1 graph images • 7 geometric images • 1 numeric data image	13 problems with image • 4 graph images • 7 geometric images • 1 matrix image • 1 numeric data image

First you can find is that many raised problems come with an image which is necessary for the full understanding of the exercise but NEVER as an additional ornament.

Study of images

From this study, a generic classification has been made, since each image has its peculiarities and can be interpretated on a bunch of different ways.

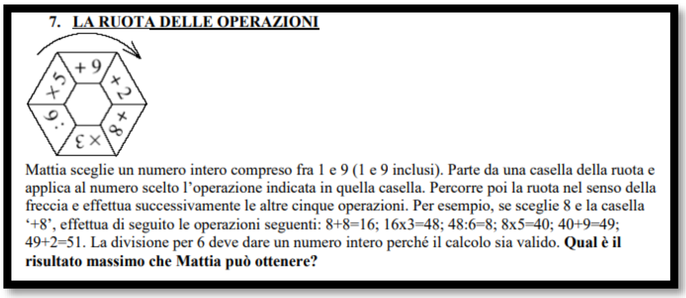
- <u>Graph images:</u> Almost every competition contains a graph which is presented in a very explicit way. That means, you can only see the nodes and roads.
  This exercises can contain just empty nodes or fill with some important numerical information. An example could be the exercise 3 of the Autumn Games.



3. <u>LA STELLA</u>

Riempite i cerchi vuoti di questa figura, in modo che ogni serie di quattro cerchi allineati sia tale che i cerchi che la compongono contengano tutti e quattro i numeri da 1 a 4.

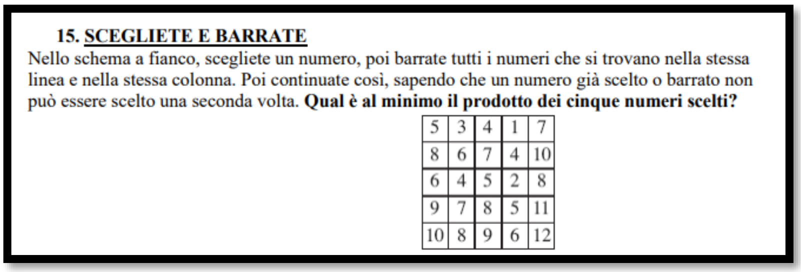
- <u>Numeric data images:</u> the most important data in this images are numbers and simbols.
  An example: exercise 7 from Autumn Games.



7. <u>LA RUOTA DELLE OPERAZIONI</u>

Mattia sceglie un numero intero compreso fra 1 e 9 (1 e 9 inclusi). Parte da una casella della ruota e applica al numero scelto l'operazione indicata in quella casella. Percorre poi la ruota nel senso della freccia e effettua successivamente le altre cinque operazioni. Per esempio, se sceglie 8 e la casella '+8', effettua di seguito le operazioni seguenti: 8+8=16; 16x3=48; 48:6=8; 8x5=40; 40+9=49; 49+2=51. La divisione per 6 deve dare un numero intero perché il calcolo sia valido. **Qual è il risultato massimo che Mattia può ottenere?**

- <u>Matrix images:</u> data is stocked inside a matrix.
  An example: exercise 15 from Autumn Games.

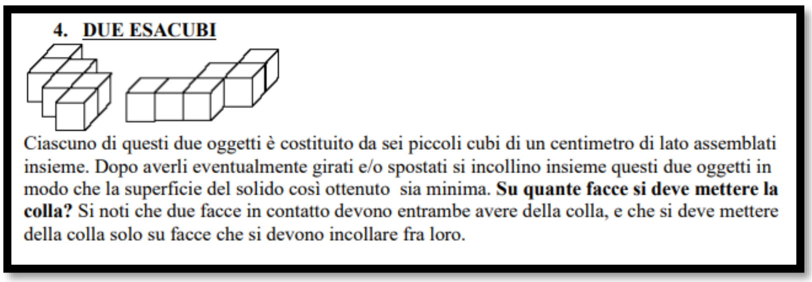


15. <u>SCEGLIETE E BARRATE</u>

Nello schema a fianco, scegliete un numero, poi barrate tutti i numeri che si trovano nella stessa linea e nella stessa colonna. Poi continuate così, sapendo che un numero già scelto o barrato non può essere scelto una seconda volta. **Qual è al minimo il prodotto dei cinque numeri scelti?**

5	3	4	1	7
8	6	7	4	10
6	4	5	2	8
9	7	8	5	11
10	8	9	6	12

- <u>Geometric images:</u> the importance of this problems relapses on related aspects of its geometry, such as: spatial understanding of ·D objects represented in 2D, extraction of geometric figures from images or identification of relationships between geometric figures and other aspects.
  An example: exercise 4 from Autumn Games.



4. <u>DUE ESACUBI</u>

Ciascuno di questi due oggetti è costituito da sei piccoli cubi di un centimetro di lato assemblati insieme. Dopo averli eventualmente girati e/o spostati si incollino insieme questi due oggetti in modo che la superficie del solido così ottenuto sia minima. **Su quante facce si deve mettere la colla?** Si noti che due facce in contatto devono entrambe avere della colla, e che si deve mettere della colla solo su facce che si devono incollare fra loro.

While the first three categories are the most easy to computationally treat and we have resourses to do it, the geometric images has a bigger difficulty to be treated in a generic way because the analysis of the characteristics are very disparate.
Worth noting that none problem uses the image as an subjective element for solving it, this means, images have no secondary meaning that could alter the meaning of the problem or the way to solve it.

Study of the statements

All the statements are brief and concise and most of al in italian in which you can find a description of a particular situation. Data is presented in a clear and explicit way, although it requires an analysis of the lenguage. Likewise, the resolution of the problem is presented in an obvious way, in the sense that it is not necessary to find hidden meanings on the statements.
In bold it is marked the main objective of the problem and it can always be find at the beginning or ending of the problem. Sometimes, the final objective comes masked as a subobjective: a concrete value is asked but it can not be extracted if the solution to the problem is not found.
If aclarations of the dynamic of a problem are made, they can be found ar the end of the statement.

<u>Example:</u>

> << Domenico utilizza un gioco da domino formato da sei tessere tutte diverse fra loro. Le due caselle di ogni tessera riportano ciascuna una delle cifre ‘0’, ‘1’ o ‘2’. Domenico ha disposto tutte le tessere sulla tavola, in modo che due caselle adiacenti di due tessere diverse riportino la stessa cifra. **Completate la figura. >>**

In this statement (Exercise 2 from Autumn Games) we can extract three types of information:

- Complete the figure: Indicates the objective of the problem.
- Game made from six pieces all different between them: elements to be completed.
- The two checkboxes from the pieces have numbers between ‘0’, ‘1’ and ‘2’: possible values that can be introduced on the problem.

Key words on objectives:

- Maximum/Minimum
- Complete.
- How many solutions…?
- Which is the value for…?

The statements change every year so there is not an regular statement structure but it is possible to find problems very similar between years.

Study of the types of problems

The problems we can find can usually be planteated from different points of view and are not anchored to a unique perspective, which means that giving them an unique category is an complicated task. As said before, the key can be the words used to define the objective of the problem.

All problems have a similar characteristic: they want to find the solution expressed on a numeric way and are not interested on the procedure.

The resource limitation is not something to worry about, at least in this case.

It is presupposed that the calculation difficulty from this problems will never be as big as the time specified on the resolution of a competition and will not use a excesive amount of memory.

We can easily see that many problems we find on the competitions can be focused from constraint satisfaction problems. The CSP are mathematical problems defined as a group of objects which state must satisfy a number of restrictions or limitations. Some of this problems can be a series of figures given in a start position and must be placed, following a number of steps, in a final position. Others can try to complete a graph or a matrix with a number of restrictions.

This problems can be solved using ecuation systems, Backtracking algorithm, search algorithm (informated or not informated) or consistency arc algorithms.

In the demo of this work we have implemented as main solution the Backtracking algorythm with Python where we compare between using or not using heuristics and if it really makes an improvement in this kind of problems. Even ore, we have solved by other methods with the objective the existence or not of an unique resolutive method.

Studied possible solution - by Backtracking

Lets start by posing a resolution by a Backtracking function. In this case, we will have to find a problem that can be solved from this method.

The problem needs to have:

- Elements where we are going to put values (graph, matrix, table or other units).
- The domain that could encompass those problems.
- The restrictions that have to be checked.
- The objective function that will say if we found a solution or not.

The pseudocode of the Backtracking function will be:

```
function BACKTRACKING-SEARCH( csp) returns a solution, or failure
   return RECURSIVE-BACKTRACKING({}, csp)
function RECURSIVE-BACKTRACKING( assignment,csp) returns a solution, or
failure
   if assignment is complete then return assignment
   var ← SELECT-UNASSIGNED-VARIABLE( Variables[csp], assignment, csp)
   for each value in ORDER-DOMAIN-VALUES(var, assignment, csp) do
      if value is consistent with assignment according to Constraints[csp] then
         add { var = value } to assignment
         result ← RECURSIVE-BACKTRACKING(assignment, csp)
         if result ≠ failue then return result
         remove { var = value } from assignment
   return failure
```

From this pseudocode, it is necessary to make some modifications to be able to choose if it is a proble with only one solution or several solutions.
In addition, to streamline the search process of the solution in case that problems were heavily big, we let the Backtracking algorithm to use or not an euristic function.
The function SELECT-UNASSIGNED-VARIABLES will return an array with the variables that are not been completed or will return an array with those variables ordered on the function by the following condition: "Which will be the most probable that will use the smallest value?" Heuristic studied and proved are summed up in the demo.

Results of the comparison between the use or not of Heuristics:

	With Heuristics	Without Heuristics
2. I DOMINO DELL'ANNO	0.00037169456481933594 seconds (average)	0.0003018379211425781 seconds (average)
3. LA STELLA	0.00038051605224609375 seconds (average)	0.0052683353424072266 seconds (average)
8. I SEI NUMERI	0.006903409957885742 seconds (average)	0.04781198501586914 seconds (average)
4. Un triangolo di numeri	0.0002923011779785156 seconds (average)	0.001165628433227539 seconds (average)
5. Una ruota di numeri	0.019980669021606445 seconds (average)	0.05500388145446777 seconds (average)
4. Vertici e somme	0.0013384819030761719 seconds (average)	0.0007157325744628906 seconds (average)

We can see a significant upgrade on some of the problems but it is not substantial based on the time scale in which we are working on. For example, in the first example, the temporal disadvantage of the heuristic is caused because there are a few elements and the domain can be a disadvantage on the use of heuristics.
The advantage of the Backtracking algorithm is that is a complete algorithm so it will always going to find, if exists, a solution or better solutions to the problem introduced.

As conclusion, Backtracking would be a good algorithm for those problems that can be arised from this perspective, but the creation of an heuristic is something really specific from a problem and, even implementing with satisfaction, it is not generalizable so heuristics are not a viable solution.

Studied possible solution - Solution by Equations or Equation Systems

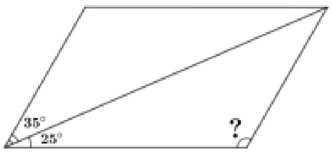
In this case, we can see that are more adapted for numeric problems of a single solution. Two types of problems are defined here:

1. Those expressed with a single equation. For example:

> **5. Il calcolo dell'anno**
> **Quale cifra va sostituita alla lettera "*a*" in modo tale che risulti corretta l'espressione:**
>
> $$\mathbf{6 \times \mathit{aaa} + 7 \times \mathit{a} = 2019}$$
>
> (*a* è dunque un numero di una cifra; *aaa* è un numero di tre cifre, uguali tra loro).

It only has to be expressed as **6 * a * 111 + 7 * a = 2019** and try to get its solution.

2. Those which come implicit on the problem. For example:

> **8. Un'eredità**
> Desiderio e Renato hanno ricevuto in eredità da uno zio d'America la proprietà di un campo che ha la forma di un parallelogramma. Nella parte del campo che spetta a Desiderio c'è un angolo (formato dalla staccionata che divide in diagonale le due proprietà) che misura 35°, in quella di Renato c'è invece un angolo che misura 25°.
>
> 
>
>
> **Quanto vale (in gradi) l'angolo di Renato che in figura è indicato con un punto interrogativo?**

For the CSP, each variable will transform into an unknown on the equation system, where the restrictions and the objective function will define it.
On the demo we have done an example of one of the two exercises done by Backtracking using a ecuation system, and a short explanation about how we get the different equations, and the system equation solving method based on matrix calculus.
The resuts shows an efficient improvement (`0.0004937648773193359 seconds`) and it also simple to develop.

Other Solutions

1. Arc-Consistency Algorithms: Even if they are algorithms focused on problems with trees where they are usually more efficient, we did not consider them because the size of the problems we are studying does not apply an improvement versus the other two methods and can even suppose a complication.
2. Search Algorithms: usually on not informated way, are a good adaption to those problems where the problem does not addapt to the CSP formalization because the modelling process is more expensive than apply this algorithms directly.

Must to mention that, in a less way, there exist problems based on logic propositional resolution, which we only found one on this three tests, and other problems based on numeric successions, count of figures, or probability counting.

> **Conclusions on resolution methods**
> The simpler the algorithm implementation, the better its adaptation to the problems.

Possible Software Archittecture

For the creation of a problem resolution software, we propose this architecture:

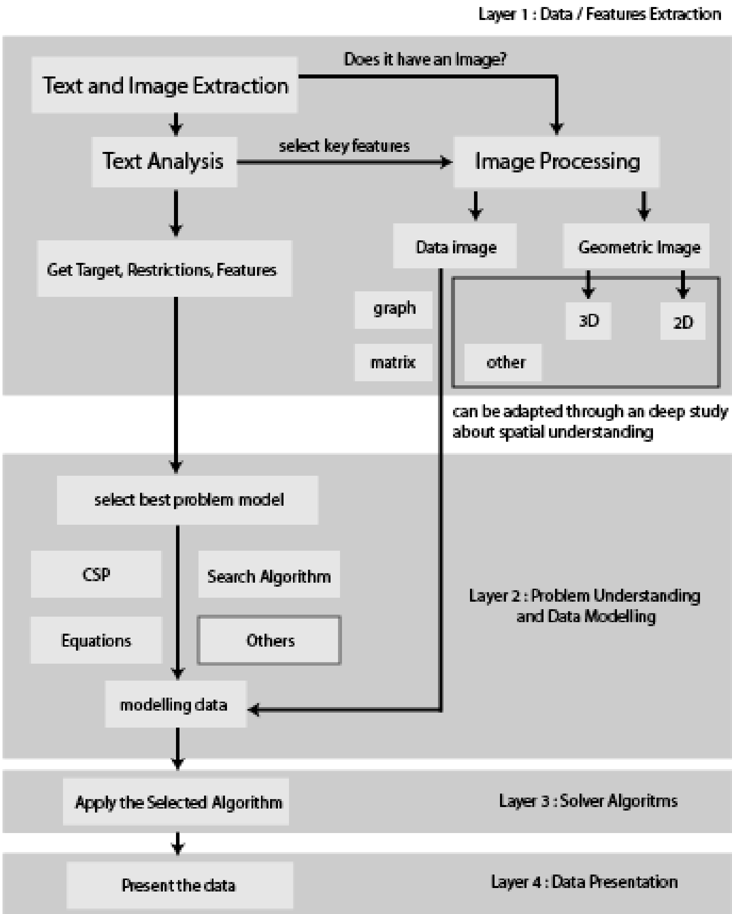


Figure 3. Architecture Schema

The true challenge on a software like this would be the modelling of the data and the problem comprehension.
The extraction of plain text and images files is very simple:

- If the data is on a PDF open file, it can be extracted directly.
- If not, an Optical Character Recognition must be used to extract the text (Tesseract) and libraries for extracting images (OpenCV).

After that, it is necessary to extract number data and key words from the statements.
Although progress has been made in this field, there is still plenty to develop (1). Here is where Deep Neural Networks are achieving the best results (2).

In the Image Processing section, firtsly we would need to identify what are we seeing (Image Clasification), that can be reforced with an input of key words extracted from the statement. This is where problems that are not represented either as matrices, as graphs, or as more or less structured numerical data are more difficult to understand, and further progress is needed in understanding the depth of flat, unlit 2D objects. (3)
After it, it is necessary to check if the problem is modelable by the programmed types ussing clasification algorithms, in case that there is not an viable algorithm, the problem is ruled out. Finally, the problem is prepared to applied the selected algorithm, processed and returns the results.

Bibliography and References

1. Natural Language Processing – State of the Art
2. https://towardsdatascience.com/deep-learning-for-specific-information-extraction-from-unstructured-texts-12c5b9dceada
3. Google AI – neural scene representation and rendering
4. Semifinale Internazionale 2018 – Bocconi
5. Semifinale Internazionale 2019 - Bocconi
6. Giochi D'autunno 2010 - Bocconi
7. https://www.zdnet.com/article/ai-aint-no-a-student-deepmind-flunks-high-school-math/
8. https://blogs.mathworks.com/deep-learning/2018/11/15/sudoku-solver-image-processing-and-deep-learning/
9. https://www.researchgate.net/publication/322583896_Deep_Neural_Solver_for_Math_Word_Problems