AN IMPROVEMENT IN LARSA AND ITS IMPLEMENTATION ON ALLOCATION OF SEATS TO CATEGORIES IN AN ORGANIZATION

Shipra Jain

Computer Science And Engineering, GGSIPU, Sector 16-C Dwarka,
New Delhi, 110078, India
shiprastays@gmail.com

Deepti Gupta

Computer Science And Engineering, GGSIPU, Sector 16-C Dwarka,
New Delhi, 110078, India
deeptigupta.1709@gmail.com

Shefali Arora

Computer Science And Engineering, GGSIPU, Sector 16-C Dwarka,
New Delhi, 110078, India
arorashef@gmail.com

Abstract

In this paper, we propose modifications to LaRSA - a fast exact algorithm for the allocation of seat for the EU Parliament. This improvement shall reduce its run time by a considerable amount. Further, we apply this modified version of LaRSA which uses the concept of degressive proportionality on the seat allocation system in Joint Entrance Examination(JEE), India. This modified algorithm takes applicants of JEE 2013 as an input and gives a fair distribution of seats among them. Our objective is to bring transparency in the system and to eliminate prejudices and political bartering.

Keywords: LaRSA; degressive proportionality; reservation ; Look Up table ; Bidirectional Traversal ; Inverse Traversal ; European Parliament

1. Introduction

The allocation of seats for the European Parliament [European Parliament Studies (2012)] is not only a significant problem, but also a scientific challenge. The European Parliament (abbreviated as **EU** Parliament or the **EP**) is the *directly elected* parliamentary institution of *the European Union* (EU). Together with the Council of the European Union (the Council) and the European Commission, it exercises the legislative function of the EU and it has been described (by its own members) as one of the most powerful legislatures in the world. The Parliament is composed of *751(previously 766) members*, who represent the second largest democratic electorate in the world (after the Parliament of India) and the largest trans-national democratic electorate in the world (375 million eligible voters in 2009). The Committee on Constitutional Affairs (AFCO) of European Parliament commissioned a Symposium of Mathematicians to ‘‘identify a mathematical formula for the distribution of seats which will be durable, transparent and impartial to politics’’. Following Grimmett (2012), the purpose was to eliminate the political bartering which has characterized the distribution of seats by enabling a smooth reallocation of seats taking into account migration, demographic shifts and the accession of new Member States.

To solve it, Janusz Łyko, Radosław Rudek of Wrocław University of Economics, proposed a fast exact algorithm, **LaRSA** which overwhelms limitations of the existing methods. It allows us to examine all feasible allocations of seats within few minutes. On this basis, an in-depth analysis of the problem is provided and some of its properties are revealed (e.g., the number of feasible allocation of seats holding the Treaty of Lisbon), which have never been presented in the scientific literature.

Furthermore, the proposed algorithm is not limited to dealing with the problem of allocation of seats for the EU Parliament, but it can be applied in the expert system for any other similar problem, especially under **degressive proportionality (abbreviated as DP)** constraints.

2. Related Work

Constraints on the Allocation method proposed in the **Treaty of Lisbon** (2007-09) by Lamassoure and Severin

(1) Maximum seats per member state (M)= 96

(2) Minimum seats per member state (m)=6

(3) Parliament size (H) = 751 , where H is the house size

(4) No smaller State shall receive more seats than a larger State,

(5) The allocation shall respect the principle of "degressive proportionality".

Cambridge Apportionment Meeting (CAM) was held at Cambridge University, on 28–29 January 2011, under the Directorship of Geoffrey Grimmett [Grimmett, G.R (2012)] for devising the mathematical formula for seat allotment. Cambridge Compromise recommendation to EU is to adopt **base+prop system:**

(1) Assign to each Member State a fixed number of seats, called the base and denoted as b.

(2) For a given divisor d, assign to a Member State with population p a further quotient p/d, resulting in the seat share b + p/d.

(3) Perform a rounding of the seat share b + p/d into a whole seat number [b + p/d].

(4) If the seat number [b + p/d] exceeds the maximum allocation, replace it by this maximum.

[Martinez-Aroza J. and Ramirez-Gonzalez V. (*2008)]* proposed some properties that a reasonable definition of degressive proportionality should possess. The paper offered a comparative analysis of several allocation methods namely **parabolic, equal+proportional, spline and power** according to such definitions with application to the EP.

The term degressive proportionality was first introduced, but not defined in the Project of the European Constitution as a rule to distribute the seats of the EP, and this continues to be such since then. The aim of degressive proportionality is to allocate to the smaller States more delegates than would correspond to their population and to the bigger states fewer delegates. A definition of DP given in 2007 by the Lisbon Intergovernmental Conference may not generally be applicable, since an allotment according to this definition may not exist for a particular composition of the EP.

*[*Ramirez-Gonzalez*, et al.(2012)]* gave a proposal to determine the distribution of seats of the EP among the member states by using **linear spline functions**. The authors studied some linear spline functions leading to allotments verifying limitations on minimum, maximum and size, and unrounded degressive proportionality (before rounding to integers). Next, restricted linear spline functions were used to obtain allotments verifying all four properties including rounded degressive proportionality, although sometimes a slightly smaller EP size must be considered.

Paolo Serafini [Serafini, P. (2012)] gave a method to deal with the problem of assigning seats to EP within the special requirements imposed by the rules of the EU using **Integer Linear Programming (ILP).** Using ILP makes central the choice of quotas to which the seats should be as close as possible. Paolo investigated how the special requirements could affect the very definition of quotas, and defined projective quotas.

In 2013, for the for the allocation of seats for the EU Parliament, researchers **Janusz Łyko, Radosław Rudek from** University of Economics, Komandorsa, Poland, developed a fast exact algorithm and named it **LaRSA** [Lyko, J., and Rudek, R. (2013)]

LaRSA overwhelmed boundaries of the existing methods and examined all feasible allocations of seats for the EU Parliament in a reasonable time, which did not exceed few minutes. Idea behind it is generation and searching of the subset for finding solution that holds degressive proportionality condition and does not exceed the seat limit.

EU problem is a multiset problem and the result has 10^{25} possible solutions which will take approximately 24 years to examine. LaRSA reduced this result set by trimming the range of solutions. It found the number of feasible allocation of seats is 195,411,484 for 2007 and 28,989,321 for 2012. These values had not been determined until then. A significant difference in the cardinality of these sets for 2007 and 2012 can be observed, whereas the demographic structure has not changed crucially.

3. Proposed method

Degressive proportionality is an approach to the allocation (between regions, states or other subdivisions) of seats in a legislature or other decision-making body. Degressive proportionality means that while the subdivisions do not each elect an equal number of members, smaller subdivisions are allocated more seats than would be allocated strictly in proportion to their population. This is an alternative to, for instance,

- Each subdivision electing the same number of members (as in the US Senate),
- Each subdivision electing a number of members strictly proportional to its population.

Degressive proportionality is intermediate between those two approaches. As a term it does not describe any one particular formula. It has following advantages:

- There may be a real or perceived danger that one or more of the largest subdivisions will dominate the legislature. This danger reduces if the votes of these subdivisions are reduced.
- The smallest subdivisions, especially those on the periphery of the territory, may have significantly different interests from many of the other subdivisions. There is a danger that these interests will be ignored if they have a tiny number of representatives. This danger reduces if their representation is increased.
- More pragmatically, the smallest subdivisions may be in a position to cause disproportionate trouble for the whole territory, for example by threatening to secede. This danger reduces if they are seen to be well-represented in the legislature.

So, DP can be applied to the problems which involve large and dynamically changing population statistics.

4. Equations

LaRSA allows us to find out that the number of feasible allocations of seats $|\prod| = 195{,}411{,}484$ for 2007 and $|\prod| = 28{,}989{,}321$ for 2012, which is a huge number thus requiring high speed computer. The hardware configuration used by Lyko and Rudek to execute LaRSA in 30 seconds is an i7, 3.4GHz processor with 8 GB RAM. The time taken to execute LaRSA on an average machine is very high. So, to reduce this time, we are proposing modifications in the original LaRSA so as to speed up the execution.

LaRSA takes n countries, where p_i denotes the population of country i for i = 1,…. .,n. For convenience, the countries are indexed according to the non-increasing order of p_i, i.e., $p_1 < p_2 <\ldots\ldots< p_n$. Each country i has assigned the number of seats s_i, where s_i is a subset { s_i^{min},…..s_i^{max} (for i = 1,. . . ,n) is the integer number, s_i^{min} and s_i^{max} are its minimal and maximal values, respectively. These values are globally bounded by the minimum m and the maximum M possible numbers of seats, i.e., $m < s_i^{min}$ and $s_i^{max} < M$ for i = 1, . . . ,n. The sum of all allocated seats (the house size) is H. Due to the Treaty of Lisbon, allocations of seats are required to satisfy a condition of degressive proportionality.

On this basis, the feasible allocation of seats (i.e., a solution) can be expressed as a tuple $S = (s_1, s_2, \ldots, s_n)$ of n elements (i.e., n-tuple), which has to hold the above constraints and $\prod$ is the set of all feasible solutions (allocation of seats), which hold the mentioned constraints.

LaRSA does not generate the total set ∏'', but at first it trims the range {s_i^{min}, . . . , s_i^{max} } for each s_i such that s_i^{min} (for i = 2, . . . ,n - 1) are the greatest possible values (but not greater than M) that hold:

$$\frac{p_1}{s_1^{min}} = \frac{p_1}{m} \leqslant \frac{p_2}{s_2^{min}} \leqslant \cdots \leqslant \frac{p_{n-1}}{s_{n-1}^{min}} \leqslant \frac{p_n}{s_n^{min}} = \frac{p_n}{M}, \quad (1)$$

whereas s_i^{max} (for i = 1, . . . ,n) are the smallest possible values (but not smaller than m) that hold:

$$\frac{p_1}{s_1^{max}} = \frac{p_1}{m} \leqslant \frac{p_2}{s_2^{max}} \leqslant \cdots \leqslant \frac{p_{n-1}}{s_{n-1}^{max}} \leqslant \frac{p_n}{s_n^{max}} = \frac{p_n}{M}. \quad (2)$$

Thus, we obtain the reduced set ∏' which is a subset of ∏''. The feasible values s_i^{min} and s_i^{max} of seats for each country holding degressive proportionality

LaRSA uses the following pruning conditions to reduce the solution set:

$$\begin{aligned} &\text{P1: } p_{k-1}/s_{k-1} > p_k/s_k \text{ for } k \geqslant 2, \\ &\text{P2: } \textstyle\sum_{i=1}^{k} s_i + \sum_{i=k+1}^{n} \max\{s_i^{min}, s_k\} > H, \\ &\text{P3: } \textstyle\sum_{i=1}^{k} s_i + \sum_{i=k+1}^{n} s_i^{max} < H. \end{aligned} \quad (3)$$

If any of them holds, then the examined subset of solutions is excluded from the further considerations

LaRSA Search tree is traversed from top to bottom, taking S_i^{min} as the initial solution.

Traversal through the LaRSA Searching Tree is done by Depth First Search algorithm to find the optimal seat allocations to the member states using a criterion function.

5. Modifications

5.1 : *Look Up Table*

It is based on Precomputation. Precomputing a set of intermediate results at the beginning of an algorithm's execution can often increase algorithmic efficiency substantially. This becomes advantageous when one or more input is constrained to a small enough range that the results can be stored in a reasonably sized block of memory. Because memory access is essentially constant in time complexity (except for caching delays), any algorithm with a component which has worse than constant efficiency over a small input range can be improved by precomputing values.

As we go deeper in the LaRSA Search Tree, the above pruning conditions are checked repeatedly for each set. In P1, we have a division operation which is very costly (as it gives floating point result) and time consuming.

The concept of look-up table table involves an array that replaces runtime computation with a simpler array indexing operation. This provides a huge boost in computation speed as LaRSA involves a large number of division operators which are both costly and time consuming.

By using this concept of look up table the time taken by LaRSA is substantially reduced as to access a value from the table the time taken is generally O(1).

	S1	**S2**	**S3**	**…**	**…**	**…**	**…**	**…**	**Sn**
P1	P1/S1								
P2		P2/S2							
…			…						
Pn				…					Pn/Sn

Table 1.1 : Look Up Table

5.2 : *Inverse traversal*

The tree obtained while implementing LaRSA is not a usual common tree. It starts with a single node (root node), grows in respect to both thickness and number of levels and finally converges into a single node again. This modification proposed by us uses the fact that the paths traversed to reach the level where the tree starts to converge are of varying distance when we approach the tree from the top and from the bottom. The number of nodes traversed may be less when the LaRSA search tree is traversed from the bottom for finding whether the current subset of solutions comply with all the constraints and are optimal and feasible solutions of the problem. The basic concept of traversing the tree is same as the one followed in case of the usual algorithm, the conditions and constraints are reversed.

5. 3: *Bidirectional traversal of the tree*

In the third modification we proposed, the LaRSA search tree is traversed from both ends i.e. top to bottom and bottom to top. Originally we were required to perform 27 iterations to find out whether the solution was feasible or not. This required a lot of time. By using this method we reduce this number by half, which means with bidirectional traversal we require only 13 iterations to find a feasible solution. In this approach we start traversing from the top-most and bottom-most node simultaneously and meet in the middle of the tree in the end.

On combining all three of these modifications we achieve a highly improved performance by the algorithm.

6. Proposed Algorithms

6.1 *Lookup Table*:

initialise_lookup

1. initialize 0th and 28th row, 0th and 97th column with 0
2. initialize lookup table, lookup[row+2][col+2]
3. for i=1 to 27, do
4. for j= 1 to 96, do
5. if (j<S_min[i] || j > S_max[i])
6. lookup[i][j] = 0
7. Else
8. lookup[i][j] = P[i]/j
9. End
10. End

6.2 *Inverse_Traversal*

Algorithm 6.2.1. [S,k] = Next(S)

1: k = n + 1 and S = (s_1, . . . , s_n)
2: For i = 1 To n
3: $s_i = s_i + 1$
4: If $s_i \leq s_i^{max}$, Then go to Step 7
5: k = n-i+1
6: End
7: For j = i-1 To 1
8: sj = sj^{min}
9: End
10: k = k - 1
11: Return [S,k]

Algorithm 6.2.2

1: Trim s_i^{min} and s_i^{max}, S = { s_i^{min};….. s_n^{min} } and k = 1
2: If P1 holds Then
3: $s_i = s_i^{max}$ for all (i = n-k, . . . ,1), [S,k] = Next(S), go to Step 21
4: End
5: If P2 holds Then
6: $s_i = s_i^{max}$ for all (i = n-k, . . . ,1), [S,k] = Next(S), go to Step 21
7: End
8: If P3 holds Then
9: $s_i = s_i^{max}$ for all (i = n-k+1, . . . ,1), [S,k] = Next(S), go to Step 21
10: End
11: If k ≠ n Then
12: k = k + 1, go to Step 2
13: End
14: If $\sum_{i=1}^{k} si_{==}$ H Then
15: ∏= ∏ U {S}
17: End
18: [S,k] = Next(S)
19: End
20: If k = = 0 Then go to Step 2
21: ∏ is the set of all feasible solutions

Algorithm 6.3: ***Bidirectional Traversal***

Algorithm 6.3.1.

[S,k] = Next(S)

1: k = n /2 + 1 and S = (s_1, . . . , s_n)

2: Take a parameter Direction such that

- Direction = 0, when traversing from top
- Direction = 1, when traversing from bottom

3: if Direction = 0 then

4:For i = n/2 To 1

5:$s_i = s_i + 1$

6: If $s_i \leq s_i^{max}$, Then go to Step 9

7:k = i

8:End

9: For j = k To n/2

10: if $s_{j-1>}$ s_j^{min} then $s_{j-1}= s_i^{min.}$

11: End

12:If Direction = 1then

13: For i = n/2+1 To n

14: $s_i = s_i + 1$

15: If $s_i \leq s_i^{max}$, Then go to Step 18

16: k = i

17: End

18: For j = i-1 To n/2+1

19: $s_j = s_j^{min}$

20: End

21: k = n-i+1

22: End

23: k = k-1

24: Return [S,k]

Algorithm 6.3.2

1: Trim s_i^{min} and s_i^{max}, S = { s_i^{min};….. s_n^{min} } and k = 1

2: If P1 holds from top Then

3: $s_i = s_i^{max}$ for all (i = k, . . . ,n/2), [S,k] = Next(S), go to Step 29

4: End

5: If P1 holds from bottom Then

6: $s_i = s_i^{max}$ for all (i = n-k, . . . ,n/2+1), [S,k] = Next(S), go to Step 29

7: End

8: If P2 holds from top Then

9: $s_i = s_i^{max}$ for all (i = k, . . . ,n/2), [S,k] = Next(S), go to Step 29

10: End

11: If P2 holds from bottom Then

12: $s_i = s_i^{max}$ for all (i = n-k, . . . ,n/2+1), [S,k] = Next(S), go to Step 29

13: End

14: If P3 holds from top Then

15: $s_i = s_i^{max}$ for all (i = k+1, . . . ,n/2), [S,k] = Next(S), go to Step 29

16: End

17: If P3 holds from bottom Then

18: $s_i = s_i^{max}$ for all (i = k+1, . . . ,n/2), [S,k] = Next(S), go to Step 29
19: End
20: If k ≠ n Then
21: k = k + 1, go to Step 2
22: End
23: If $\sum^{k}_{i=1} si_{==}$ H Then
24: ∏= ∏ U {S}
25: End
26: [S,k] = Next(S)
27: End
28: If k = = 0 Then go to Step 2
29: ∏ is the set of all feasible solutions

7. Performance details and Result

For performance analysis we decided to run the program for same amount of time and compare the results obtained. The time considered for both approaches was 3 min 19 seconds**.**

a) **The input figures were taken from the population statistics of member nations of European Union (2007).**

<u>CPU time taken :</u>

In case of the original LaRSA ,
CPU time taken was 1 min 52 sec (outside kernel)
And 0 min 3 sec (inside kernel)
Therefore total CPU TIME = 1 min 55 sec

In case of modified LaRSA,
CPU time taken was 1 min 23 sec (outside kernel)
And 0 min 6 sec (inside kernel)
Therefore total CPU TIME = 1 min 29 sec

<u>Number of Outputs produced :</u>

In case of the original LaRSA
Solutions produced = 10655
In case of modified LaRSA,
Solutions produced = 203482

b) Performance on Reservation data

The input taken for this analysis was population data of jee 2013. The common time considered for both approaches is 4 min 50 seconds

<u>CPU time taken :</u>

In case of the original LaRSA ,
CPU time taken was 1 min 47 sec (outside kernel)
And 0 min 25 sec (inside kernel)
Therefore total CPU TIME = 2 min 12 sec

In case of modified LaRSA,
CPU time taken was 1 min 55 sec (outside kernel)
And 0 min 9 sec (inside kernel)
Therefore total CPU TIME = 2 min 4 sec

Number of Outputs produced :

In case of the original LaRSA

Solutions produced = 29131

In case of modified LaRSA,

Solutions produced = 100401

For EU Parliament, the performance analysis clearly shows that the CPU time taken by **modified LaRSA is less than that by original LaRSA.** We can see how the modified LaRSA produces more output. **The modified LaRSA is 94.76 times more solutions than the original LaRSA.**

Modified LaRSA is implemented on the reservation problem successfully, taking populations of each category who appeared for JEE 2013 as an input. Output obtained are the no of seats allocated to each category. Allocation of seats satisfies degressive proportionality and provides a intermediate solution between equality and proportionality approaches.

When we applied the original LaRSA algorithm on the JEE statistics, 29131 feasible solutions are obtained in 4 min 50 seconds, while our modified version of LaRSA gives 100401 solutions in the same duration, Hence, **modified LaRSA produces 70.98 times more solutions than the original LaRSA.**

8. Conclusion and Future Work

To improve the performance of LaRSA algorithm, three different modifications were proposed namely look up table, inverse traversal and bidirectional traversal of the search tree.Lookup table used in LaRSA avoids repetitive computation. The results show that the program executes faster using the lookup table generating more feasible solutions in a unit of time than the original LaRSA.

We then thought what problem can use this concept for provision of a solution. We came up with the problem of reservation of seats in various organizations and educational institutes in our country where we could apply degressive proportionality to gain a more practical and transparent solution as both EU parliament problem and reservation problem in India involves large population data which is usually dynamic in nature.

In the future, this modified LaRSA can be further revised to take into account various factors which may affect reservation like income, literacy, area or region of settlement etc. This can be achieved by mining the population data according to these factors by using appropriate data mining tools.

9. Acknowledgements

The authors would like to thank their faculty advisor Dr. Devendra Tayal, Associate Professor and ex-HOD, Indira Gandhi Institute of Technology (now IGDTUW), Delhi, to support this work.

References

[1] Blanca L Delgado-Márquez, Faculty of Economics and Business, University of Granada,(2013). A more balanced composition of the European Parliament with degressive proportionality

[2] Dniestrzański, P. (2011a). Diversification of distribution of seats in European Parliament (Dywersyfikacja podziału mandatów w Parlamencie Europejskim). *Ekonometria*. *33*, 32-41. Dniestrzański, P. (2011b). Degressive proportionality – source, findings and discussion of Cambridge Compromise. *Mathematical Economics, 6 (13)*, 39-50.

[3] Dniestrzański, P. (2013). The proposal of allocation of seats in the European Parliament – the shifted root. *Procedia – Social and Behavioral Sciences (*in press).

[4] European Parliament Studies PE 432.760. <http://tinyurl.com/6f3ff6u> [accessed: 2012-06-14].

[5] Florek, J. (2012). A numerical method to determine a degressive proportional distribution of seats in the European Parliament. *Mathematical Social Sciences, 63 (2)*, 121-129. http://dx.doi.org/10.1016/j.mathsocsci.2011.07.003.

[6] Grimmett, G. R. (2012). European apportionment via the Cambridge compromise. Mathematical Social Science, 63, 68–73.

[7] Grimmett, G. R., Laslier, J. -F., Pukelsheim, F., Ramírez González, V., Rose, R., Słomczyn´ski, W., Zachariasen, M. & Zyczkowski, K. (2011). The allocation between the EU Member States of the seats in the European Parliament.

[8] Introduction to algorithms by Thomas H. Cormen. Charles E. Leiserson. Ronald L. Rivest. Clifford Stein , second edition, The MIT Press , chapter 23 , page 540

[9] Łyko, J., & Rudek, R. (2013): A fast exact algorithm for the allocation of seats for the EU Parliament. *Expert Systems with Applications, 40 (13)*, 5284-529.

[10] Martinez-Aroza, J.; Ramirez-Gonzalez, V. (2008): Several methods for degressively proportional allotments. A case study. Mathematical and Computer Modelling, 48,1439–1445.

[11] Pijils, W., Post, H. (2006). A new bidirectional search algorithm with shortened post-processing. European Journal of Operational Research.

[12] Ramirez-Gonzalez, V., Martinez-Aroza, J., & Marquez-Garcia, A. (2012): Spline methods for degressive proportionality in the composition of the European parliament. Mathematical Social Sciences, 63, 114–120.

[13] Serafini, P. (2012). Allocation of the EU Parliament seats via integer linear programming and revised quotas. Mathematical Social Science, 63, 107–113.

[14] Wen-Chiung, L. ; Shiuan-kang C. ; Chin-Chia W., Branch-and-bound and simulated annealing algorithms for a two-agent scheduling proble Department of Statistics, Feng Chia University, Taichung, Taiwan.