Implementing the General Theory for Finding the Lightest Manmade Structures Using Voronoi and Delaunay

Mohammed Ezzat¹
¹German University in Cairo, Cairo, Egypt
¹mohammed.ezzat@guc.edu.eg

In previous efforts, the foundation of a general theory that searches for finding lightest manmade structures using the Delaunay diagram or its dual the Voronoi diagram was set (Ezzat, 2016). That foundation rests on using a simple and computationally cheap Centroid method. The simple Centroid method is expected to play a crucial role in the more sophisticated general theory. The Centroid method was simply about classifying a cloud of points that represents specific load case/s stresses on any object. That classification keeps changing using mathematical functions until optimal structures are found. The point cloud then is classified into different smaller points’ groups; each of these groups was represented by a single positional point that is related to the points' group mean. Those representational points were used to generate the Delaunay or Voronoi diagrams, which are tested structurally to prove or disprove the optimality of the classification. There was not a single optimized classification out of that process but rather a family of them. The point cloud was the input to the centroid structural optimization, and the family of the optimized centroid method is the input to our proposed implementation of the general theory (see Figure 1). The centroid method produced promising optimized structures that performed from five to ten times better than the other tested variations. The centroid method was implemented using the two structural plugins of Millipede and Karmaba, which run under the environment of the Grasshopper plugin. The optimization itself is done using the grasshopper's component of Galapagos.

Keywords: Agent-based structural optimization, Evolutionary conceptual tree representation, Heuristic structural knowledge acquisition, Centroid structural classification optimization method

MODEL DEFINITION:
The general theory defined in the backing paper (Ezzat, 2016) was developed based on the need for having a better understanding and explanation of the point cloud. Simple unanswered, but important, questions like the representational question of what if each classified points’ groups to be represented by a curve, a surface, or a mass instead of just a sin-
The different centroid optimizations

The presumed general theory optimizations

The family of optimized centroid Voronoi/Delaunay diagrams as inputs of our proposed general theory optimizations.

gle positional point needed to be answered. Our proposed implementation of the general theory is a skeptic approach. Our proposed model is a skepticism of the centroid method achievements. The model would lay an algorithmic framework for any viable further specific implementation. That framework would guaranty the computational optimality and would assure the viability of answering any pool of questions, including the original representational question. The framework is presumed to give us the optimum needed understanding of the point cloud.

The original paper introduced the concept of zooming-in-zooming-out. After exhausting many of the other available possibilities, we were faced with the reality that the zoom-in-zoom-out technique is the best approach for implementing our model. Actually, the success of the whole model rests on that simple concept. As it would be clear soon, the model’s Big-O Algorithmic Complexity is \( (\Theta(1) \rightarrow \Theta(x)) \) for all the different processing scenarios is due to that concept (Vrajitoru and Knight 2014, Skiena 2012). Figure 3 contextually represents our proposed model in relation to the centroid method. The family of optimized centroids, the set \( I \) where each \( I_n \in I \) for \( n \in \mathbb{Z} \) (the positive integers), is the initial input of the model. The model is composed mainly of two processes. The first is the analytical process, while the second is the behavioral one. Both of these processes keep communicating, and in certain occasions, the centroid model is used to assure the correctness of the conclusions and/or to enrich the inputs’ set with new updated ones. Both of the processes are structured based on the zoom-in-zoom-out concept. The zoom-out represents the summary of the data, the zoom-in represents the detailed analysis, while the optimized version is located in between (see Figure 2). Figure 2 represents our “conceptual tree”, which would link the two analytical and behavioral processes. It will also facilitate their communication and assure the computational superiority of any further specific implementer. The continuous calculative communications between the analytical and the behavioral processes are done in parallel with the centroid structural tests and optimizations, which are done only when needed because of their relatively costly calculations. These continuous communications are meant to adjust the analytical conceptual lattices themselves and to propose physical variations by the aid of the behavioral agents.

The following procedures are considered as influential computational measurements:

- The searching deep in the “conceptual tree” is always based on the criterion of desperate need.
- Any possible searching duplications are avoided, meaning that the search would not reach the deeper nodes of the tree unless that node is unique and has no other similar nodes in the tree.
- The focus would always be kept as possible on the roots of the tree. The data flow from the
The analytical process:
The conceptual analytical data is represented mathematically as related lattices (see Figure 4) (Kaburlasos 2006, Gen and Cheng 2008, Sierksma and Ghosh 2010, Parrochia and Neuville 2013). Lattices are Partially ordered sets coupled with the $\text{Inf}(\wedge)$ and $\text{Sup}(\vee)$ operators. Consequently, to represent the analytical data using lattices, the partial ordering relation $\leq_{ab}$ and the two $\{\wedge, \vee\}$ operators need to be defined for any interpretation of the analytical data’s variations. Other parallel definitions of lattices could also be used when needed. Therefore, the possibility of using the mathematics of the lattice theory to represent the acquired or retracted knowledge of data, actions, or the interactions between them is a needed arsenal during the optimization (Gratzer 1998). The concept theory is an exemplary of using lattices to represent knowledge (Ganter and Wille 1996, Ganter and Obiedkov 2016). Each node of the “conceptual tree” has one or many lat-
tices that represent the different notions related to each zooming-scale (see Figure 4). Later In the paper, a discussion of the ordered relations $\leq_{ab}$, coupled with the $\text{Inf}(\wedge)$ and $\text{Sup}(\lor)$ operators to represent the model’s knowledge repository of each zooming-scale would be conducted. The “conceptual tree” is adaptive, meaning that it may grow in specific locations only if needed. Proposed techniques for the analytical data storage in the “conceptual tree” would be discussed (see Figure 4).

The first and the second tasks are related to the maturity phase, while the “conceptual tree” modifications are possible in the optimization phase. The behavioral process is done by different agents’ families. Each agent family has a certain action to perform. The different agents’ families are to construct a communicative society. No matter the agents are strictly structured or loosely structured, they are always allocated to the proper “conceptual tree” nodes as needed (see Figure 5). The agent’s families, algorithmically, mainly execute the aforementioned behavioral, inferential, or optimization tasks.

**DISCUSSION:**

The proposed framework’s efficiency is founded on two theoretical premises. The first premise is that the framework’s algorithmic performance is fast enough, while the second premise is that the framework is a needed practical infrastructure for the specific implementer. The algorithmic first premise could be easily theatrically proved, the model’s Big-O Algorithmic Complexity is $\Theta(1) \rightarrow \Theta(x)$. The second practical premise is the one that we may have a hard time trying to prove. One of the reasons of that hardship is the lack of the existence of a specific practical implementer. The existence of a specific implementer may help in assessing the framework’s practicality by comparing its structural optimization results against the counter centroid method’s achieved results. In the next section, we would try to define a possible specific implementer and then we would engage in a qualitative analytical discussion about the practicality and the necessity of the framework. Finally, an analysis of the general theory in the context of structural optimization would be looked over by the end of this section.

Figure 3
Our proposed framework’s analytic and behavioral processes in relation to the centroid optimization engine. The inputs of the two models are as illustrated. Our proposed optimization as structured non-monotonic heuristic search of optimal variation/s from the infinite design space.
the different lattices \( L_{i\ j} \) allocated at each input tree \( I_a \) nodes \( i_{ab} \) as needed. The three zooming-scales are analyzed by these lattices with the appropriate ordering relations.

**An exemplary specific implementer:**

The specific implementer would start the journey of having a comprehensive epistemological understanding of the point cloud by answering a pool of questions. That pool of questions, including the representational question, may be considered as the possible known variations tested over the possible known Voronoi/Delaunay cells, which represent the design space. There is a simple choice to make, either to test these design space variations and their consequences thoroughly, inapplicable brute-force like, or else to rely on the framework heuristic non-monotonic reasoning during the epistemological journey (see Figures 3 and 6). The point cloud’s knowledge acquisition relies on answering the pool of questions using the family of the centroid optimizations. The knowledge acquisition would be done over any of the three initial, maturity, or optimization phases.

Therefore, in the initial phase, the first thing to do is to define the conceptual tree/s using the family of the centroid optimizations \( I_n \). The best optimal will be used to cheaply and speedily define the tree’s three zooming scales (see Figure 6). The first zoning is defined using the opposite classification curve and then for each of the classified grouping of each zone, a further classification using the same curve is conducted and so. The number of the clusters of each zone is defined by the specific implementer’s interpretations. Consequently, the conceptual tree \( T_n \) is constructed from the best optimal centroid \( I_n \). This elementary form of the conceptual tree would evolve alternatively over the three phases until *optimality in conceived*. The rest of the family of the centroid optimizations would be mainly analyzed based on their discovered classification curves. The definition of the conceptual tree is an essential mandate of the initial phase, and it should be coupled with initially defined analytical and behavioral conceptual lattices.

The creation of these initial lattices could be done by the following:

- The agents at this phase would allocate themselves, or recommend to be allocated, on the proper behavioral lattices \( L_{ab} \) on the proper tree’s nodes \( N_n \). Optimally speaking, a priori knowledge structure could propose default lattices and then the different agents would subscribe consequently.
• The lattices are finite and hence there should be a $[0, 1]$ for each lattice. Each agent could first allocate itself to the proper behavioral lattices then iteratively they would relate themselves to other agents using the ordering relation $\leq_{ab}$ or the $\{\land, \lor\}$ operators.
• The initially concluded lattices would be defined based on the $[0, 1]$ assumption and the proposed relational operators.

The consequent maturity phase conducts the heuristic search for optimal Voronoi/Delaunay diagrams’ variations, but without changing the corresponding conceptual tree. This phase is mostly a learning phase; the hypotheses of optimal diagrams over the possibilities’ space are more regularly attested using the centroid method. These empirical knowledge acquisition feedbacks are of great help not only to this phase but for the coming phase as well. The agents that may heuristically modify the diagrams by adding, deleting, or modifying certain points in specific locations are the main acting agents in this phase. These altered points always belong to the same node of the conceptual tree. Some actions at that phase may include the following:

• The recommendations of the previous phase are taken into considerations. A higher authority, either elected by the agents or defined by a priori knowledge, would allocate the agents on the proper lattices on the proper tree’s nodes.

• Both the analytical and the behavioral processes will be fully conducted in this phase. Creation and alterations of all the lattices, the agents’ allocations or generations, and the integration between the two behavioral and analytical processes are the main activities that would happen in this phase. This does not include the agents that would modify the conceptual tree, but it would include the agents that may modify the corresponding physical Voronoi/Delaunay diagrams. The main goal of this phase is to epistemologically examine the possibilities of the design space of the corresponding diagrams using a single given unaltered conceptual tree.

• Recommendations for the agents that modify the conceptual tree are proposed for the next phase to take care of. The next phase “conceptual tree” modifying agents may include the agents that expand, contract, or redefine the clustered point cloud zooming-scales.

In this last optimization emergence phase, The agent-families have the full control and the emergence of the conceptual tree and/or its corresponding diagrams are up to the limits. The empirical optimizations of the previous phase are taken into considerations. For the sake of predictability, no huge modifications of the corresponding diagrams beyond the assertions of the previous phase should take place. In the case of massive modifications of
the corresponding diagrams are assumed, a transition from this phase to the maturity phase should take place.

• This phase concludes by a reformed “conceptual tree”. Aged agents are defined by the end of this phase. Sometimes, the transitioning may happen to the initial phase rather than the maturity phase in the case of major changes in the conceptual tree itself. The already developed lattices have better chances of survival.

The practicality of the framework:
The exemplary specific implementer is an agent-based solution that epistemologically exhausts the infinite design space variations using non-monotonic heuristic uncertain searching for global optimal conceptual tree/s, and their corresponding diagram/s (see Figures 3 and 6) (Rothlauf 2011). In other words, this is not an environment simulated by agents but rather it is a goal-driven process. The specific implementers’ Adaptive agents are pieces of software that are autonomous, have brains, cooperative, share the same goal, and asynchronous (Barbucha, et al. 2013). For this multi-agents’ epistemological solution to exist, infrastructural Architectures are needed to produce qualitative optimal solutions by establishing a platform for the agents’ collaboration and communication. This infrastructural architecture is the proposed framework. The agents’ non-monotonic heuristic reasoning is meant to discover rules and to define concepts (Resconi and Jain 2004). The adaptive agents are hierarchically structured over the conceptual tree nodes, and the conceptual tree is the most important element of the proposed framework. This nodal structure implies the existence of hierarchical contexts over the various zooming scales. The agents make their rules’ or concepts’ inferences based either on their internal contexts, the nodal context they occupy, or on the external nodal contexts, either higher or lower in the hierarchy (Resconi and Jain 2004). Concepts are suitably represented and processed using the lattices’ mathematics (Ganter and Obiedkov 2016). For the agents to discover rules, they need to be intelligent. Their internal architecture is adjustable or neurally defined at the run-time, and they could use the AI machine learning techniques to empirically make their epistemological inferences (Smajgl and Barreteau 2014). Their intelligence is not limited to their internal architectural adaptability, the main goal of defining the emergence of the behavioral lattices is to define various convergence schemes between these agents. These various convergences are to guarantee rich combined intelligent behaviors between the suitable agents.

The conceptual tree is a representation of our in-depth understanding of the point cloud, this understanding simply incepts from the optimized classification bell-shaped curve of the centroid method and gets evolved and developed until plausible optimal understanding of the point cloud is discovered (see Figure 6). The life cycle of the conceptual tree keeps evolving over the alternating three states; and ends by conceiving optimal understandings. The second maturity phase is full of the potentialities for learning, and the analytical centroid method is mostly visited to assert hypotheses at that phase. All the behavioral/analytical activities are conducted in that phase, except the modification behaviors of the conceptual tree itself. The agents at different levels would have a synergic non-monotonic decision of which variations are eligible for the centroid method to examine and which are exempted from further analysis. In the last optimization phase, the conceptual tree modifications are viable based on the learnt experiences from the previous maturity phase. A decision is taken afterward to transit the conceptual tree back into one of the earlier two phases, mostly the second phase unless major modifications are expected in the conceptual tree or the corresponding diagrams. The agents should act according to each phase functionality; they collectively, electively or combined with a priori controllers, should define their aging schemes, the dependability on the nodal internal or external in-
ferences, and the conceptual tree’s *transitioning times between the phases*. The computational and functional superiority of the conceptual tree is more obvious by now. Nevertheless, the conceptual tree may have two possible variations by the implementers, both of which would still support the necessity of the conceptual tree:

1. The conceptual tree may ultimately transform into any other homomorphic graph type that is favored by the specific implementer.
2. There could be other favored parallel graph types during the analytical evolution of the conceptual tree that link the tree’s nodes differently (Chein and Mugnier 2009).

Lattices and their capabilities of representing concepts, which are intrinsically hierarchical, are of great support for the conceptual tree’s *evolutional optimization*. The analytical and behavioral lattices would always entail that ordered *evolutional optimization* of the conceptual tree and would have a positive contribution to memory management.

Although the design space variations are virtually continuous, plenty of discrete variations, the main goal of the general theory is to transform the virtually continuous solution space into a combinatorial one using the hierarchical property of the conceptual tree. The hierarchical structure of the conceptual tree should aid in the dynamic evolving of the infinite population of solutions of the design space. The agents should heuristically *elevate* or *remove* solutions from the population during the conceptual tree’s *evolution* over the *alternating three phases*. That addition and removal of the solutions’ population should ultimately culminate into fewer possible optimal conceptual trees and their corresponding diagrams (Barbucha, et al. 2013). The role of the questions’ pool is to prioritize the variations that need analyses or examinations during that population’s selectivity process. We may now mention two main criteria for evaluating the successfulness of the specific implementer as of the following:

1. The less the amount of revisions needed by the centroid method during the whole optimization process the better the specific implementer is.
2. The quality of memory management and the ability to integrate it with the evolving rules and concepts of the framework, to keep it probabilistically reliable and small as much as possible, is of high importance.

The optimal memory utilization and migration during the general theory optimization can mostly be achieved using rules rather than using declarative reserved memorial spaces. This can be achieved by learning proper rules for managing the centroid calculations. Rule-base memory managements are probabilistic and repeating centroid calculations of same cases is possible.

The last point to discuss would be to clarify some of the reasons behind selecting this bottom-up distributed decentralized multi-agents’ searching solution over the centralized monotonic deductive approaches (Weyns 2010). The autonomous intelligent multi-agents’ solution is more suitable to the complex optimizations, like ours. The suitability of that approach may be clearer from the eyes of Software development. Although the multi-agents’ solution would require a developed core to build upon, that is the modules of our proposed infrastructural *framework*, the multi-agents systems are always computationally more efficient, reliable, and maintainable. But most importantly, it is extensible (Barbucha, et al. 2013). This extensibility property would prove to be of primal importance to our optimization. The proposed agents’ families in the paper are exemplary and are not meant to be exhaustive by all the means. Actually, this is the main task the specific implementer should tackle. The specific implementer may be in need for specific optimizers or any other objects, it will only be needed to code that simple agent’s piece of software and then extending the solution by simply plugging new agent in it. The speed of the algorithmic optimization engine is expected to grow over time, based on the built-in learning capabilities of the agents.
The general theory in the context of structural optimization:

The field of structural optimization flourished during the previous decade. The three main applications of structural optimization and control are topology optimization, shape optimization, and size optimization (Bendsøe and Sigmund 2004). Comparing these applications with the general theory would give us a better understanding of the general theory in the context of structural optimization. Size optimization is interested in optimizing the sizes of the already optimized layouts. Size optimization is an integrative part of the centroid method itself, and it is done using the same classification algorithms of the centroid method. Physically wise, size optimization enhanced the structural performance of the optimized Voronoi/Delaunay diagrams from two to three times. On the other hand, the centroid method's size optimization is considered as a paired algorithm with that of the centroid method's classification optimization. As an explanation of that algorithmic coupling of the two algorithms, the centroid method's classifications are done against the single Deflection criterion, the chords' structural utilization criterion is not included in that centroid's optimization, that is because the centroid method is meant to be fast and computationally cheap for the general theory possibility be built on it. The size optimization, as a coupling algorithm with the centroid method, may give us a further insight of which chords that could be deleted, and hence a proposed consequent rephrased final diagram is viable.

Nevertheless, the most important structural optimization to compare against the general theory is topology optimization (Bendsøe and Sigmund 2004). Topology optimization searches for the optimal layout of the material in the design domain. That design domain is very much similar to the point cloud of the proposed general theory. We would try to relate the general theory to topology optimization using the following comparative or integrative methods:

- The comparative method: topology optimizations are mostly 3d free forms that can be produced using Additive industrialization, 3d printing, while the general theory Voronoi/Delaunay diagrams produce straight lines, which may be industrialized using the standard profiles cut to the needed lengths. A deflection criterion comparison between the two theories, using the same mass of materials under the same load case/s, is viable using the specific implementers of the theory, which are viable using the centroid method and the proposed framework.
- The integrative method: the possibilities of integrating the analytical techniques of the two theories could be investigated. The epistemological approach of the general theory could be applied to the topology optimization applications. The topology optimized structures could be integrated with the specific implementers in structuring the questions' pool or in prioritizing the agents' efforts in spanning the infinite design space.

Shape optimization is another viable, possibly parallel, application of the general theory. The general theory could be merely a shape optimizer of an optimized centroid diagram; the centroid method and the proposed framework would then be utilized by the similar techniques introduced in the paper. In this case, the general theory could produce free formed optimizations or other geometric forms of the already optimized diagrams' chords (Akbarzadeh, Mele and Block 2015). These aforementioned variations are expected to participate positively in the architectural-structural various integrative scenarios.

CONCLUSION:

We propose an indispensable complementary framework of the general theory. It does not matter how sophisticated or successful the communicative, behavioral, or analytical algorithms used by the specific implementations are, our framework would always persist as a factual implication. Therefore, the framework needs serious comprehension and elab-
oration. There are two main benefits of the framework. The first benefit is the epistemological parallelism between the behavioral/analytical modules and the centroid analyzer/optimizer, while the second benefit is the framework’s Big-O Algorithmic Complexity of \( O(1) \rightarrow \Theta(x) \). Our understanding of the point cloud is represented using the conceptual three that initializes based on the defined classifying curve of the centroid method’s optimal Voronoi/Delaunay’s diagram. To have a comprehensive understanding of the point cloud, we need to discover optimal conceptual trees and their corresponding diagrams. The initialized conceptual tree would keep heuristically evolving over the alternating three phases until optimality is conceived. This heuristic non-monotonic probabilistic search of optimality is fast and suits the complexity of the problem in hand; it is done using adaptable intelligent multi-agents’ solution that builds on the proposed framework. For our future work, specific implementers would be defined and their structural optimization's deflection results would be compared to that of the centroid method and to that of other structural optimization theories. Integrative analysis between the specific implementer and other structural optimization theories would be practically investigated too.

REFERENCES


Chein, Michel and Mugnier, Marie-Laure 2009, Graph-based Knowledge Representation: Computational Foundations of Conceptual Graphs, Springer-Verlag,

London


Ganter, Bernhard and Obiedkov, Sergei 2016, Conceptual Exploration, Springer-Verlag, Berlin, Heidelberg


Gratzer, George 1998, General Lattice Theory, Birkhauser Verlag, Basel, Boston, Berlin


