Visual Exploration System in an Industrial Context

Andrew Fish *Member, IEEE*, Claudio Gargiulo, Delfina Malandrino, Donato Pirozzi, and Vittorio Scarano

***Abstract*—This paper describes ExploraTool a new interactive tool to visually explore data from multiple repositories. The tool has been applied in a real setting to explore CFD simulation data and obtain new insights into the space of simulations. The inclusion of free exploration, filtering operations and chart generation provides a quick method for performance comparisons. The paper proposes an algorithmic means of processing input in the form of tabular data sets, generating a plausible hierarchical structure over metadata categories which is used to initialize the visualisation together with interactions methods to explore, select and compare sets of simulation data. This paper also reports on the Evaluation Study performed involving 24 engineers over two distinct locations from a large automotive manufacturer, to evaluate the usability and the overall user satisfaction with the tool. Participants rated the tool as intuitive, useful and effective.**

***Index Terms*—Exploratory Search System, Information Retrieval, Data Visualisation, User Interfaces, Multirun Simulations, Industrial User Evaluation Study, Dataset processing.**

I. Introduction

With the increased availability of computing power and storage capacity, medium and large enterprises can continuously collect data along the product design process (PDP). Enterprises have large, or even big, data repositories of potentially valuable and strategic assets. In order to boost their competitiveness it is becoming vital to obtain insights into the repositories, exploring their content and extracting new knowledge. Exploring repositories to find valuable information is difficult [1] because data management systems use traditional list based widgets, displaying only a small data portion compared to the repository size, so researchers are exploring the use of other visualisations (e.g., treemap). Without an adequate exploring system, data remains in the repositories without exploitation.

Automotive manufacturers have multiple repositories in which to store: simulation data generated by different simulator software, experimental data continuously collected from the Wind Tunnel infrastructure, and competitors' product data performances accessible through subscription to third part services (e.g., A2Mac1). These repositories are independent, and in order to compare the various data sets across repositories, analysts often have to manually access each of them. Due to this scenario, and market competition, an increasing desire to facilitate easy exploration of repositories, select groups of data, aggregate them, and perform comparisons over the data via an intuitive interface is evident. In contrast, software engineering companies often focus only on the simulation functionality (e.g., Computer Aided Engineering functionality), not addressing the actual industrial need to explore their repositories, and compare data with other simulators' results.

This paper introduces a web-based tool, called ExploraTool that enables the visual and interactive exploration of data sets by item properties. Using ExploraTool, analysts can select multiple groups of simulations or single simulations, and compare their relevant simulations' performances.

ExploraTool has multiple benefits: 1) it provides a visual overview of the repository content, grouping items together by their properties (facets) and visualising them using nested ellipses covering all the available 2D space screen; 2) compared to the traditional list-based results, during the navigation it gives a clue on the overall repositories content, which would be non-trivial with traditional file systems organised by directories and not by item properties; 3) the visualisation based on facets helps the user to explore and discover item properties to further investigate and filter the items by selecting ellipses; 4) ExploraTool reads data from multiple repositories (i.e., multiple network file systems, external sources), allowing the Analysts to select simulations from different sources; 5) it is extensible to be able to also explore experimental data and competitors' performance data, integrating together different data sources to have an all-in-one workbench.

The paper is organised as follows. Section II discusses the related work in terms of Exploratory Search Systems and visualisation approaches. Then, the paper evolves and combines four aspects that are the main paper contributions: 1) the ExploraTool idea, its graphical user interface and features described in Section III; 2) the mathematical background and an algorithm to process as input any kind of tabular data set and impose on it a plausible hierarchical structure over metadata categories that ExploraTool is able to visualise, enabling data set exploration, described in Section IV; 3) a field study performed within a large automotive manufacturer, which involved 24 engineers, to evaluate the tool usability and overall user satisfaction, which is essential for industrial adoption described in Section V; 4) an overview the future extensions obtained through both the field evaluation and the stakeholders' interviews, along with lessons learnt from the process and the potential usage of the tool and approach in other industrial sectors, described in Section VI.

A. Fish is with the School of Computing, Engineering and Mathematics, University of Brighton, UK (e-mail: Andrew.Fish@brighton.ac.uk)
C. Gargiulo is with FCA (e-mail: claudio.gargiulo@fcagroup.com)
D. Malandrino, D. Pirozzi and V. Scarano are with the Department of Computer Science, University of Salerno, IT (e-mails: dmalandrino@unisa.it, dpirozzi@unisa.it, vitsca@dia.unisa.it)

II. Related Work

This paper focuses on the exploration of industrial repositories that store thousands of simulations performed over years.

It aims to assist the industrial analysts with their need to gain insight, understand the content of the repositories and filter data to help perform comparisons. Instead of displaying a list of results, this work exploits the use of interactive visualisation to provide a visual indication of the repositories' contents, whilst supporting data filtering and selection tasks. Hence, exploratory search systems, visualisation approaches, and interaction techniques are related.

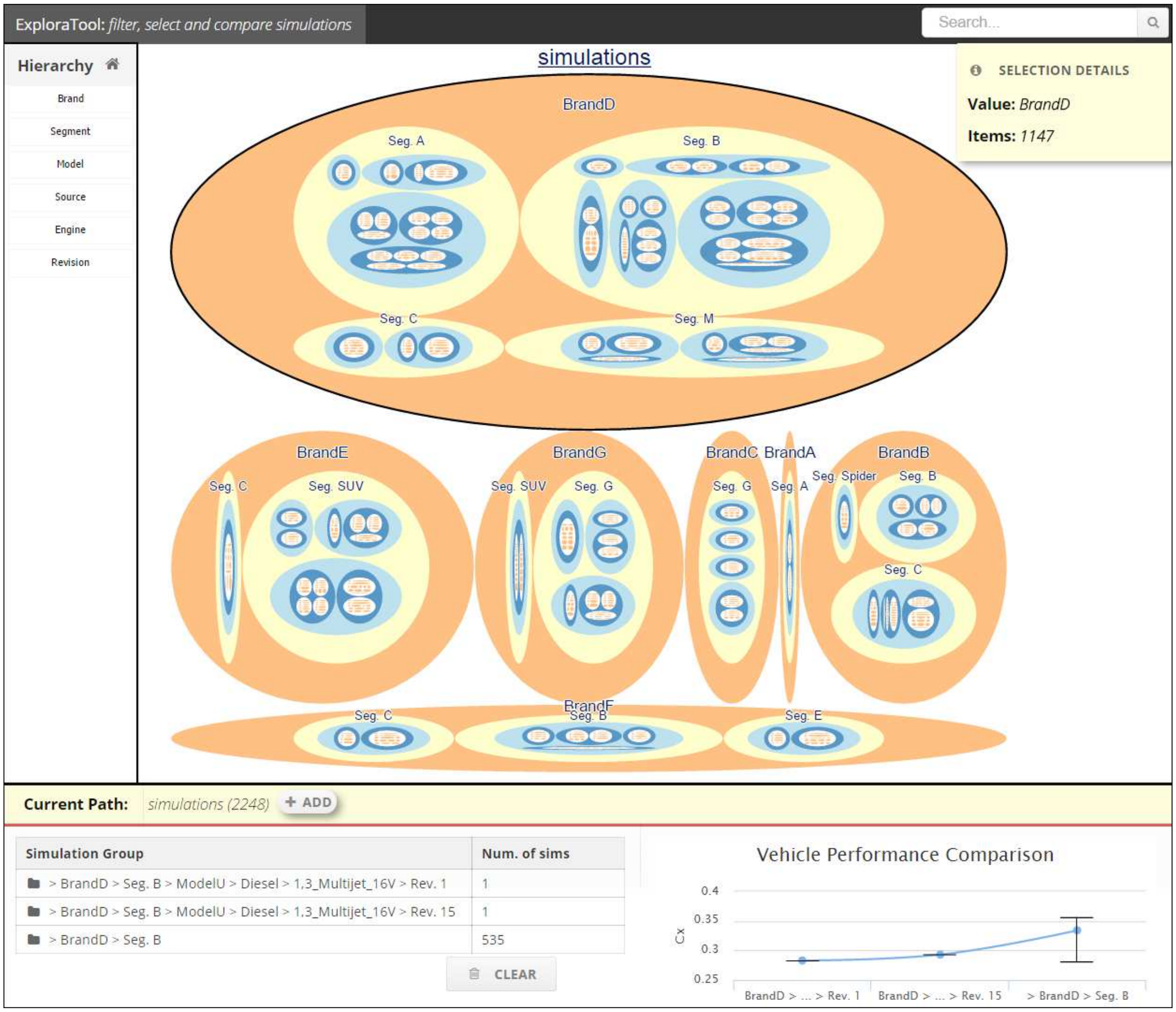
Classical *lookup search* [2], [3] is a query-response paradigm where the user poses a textual query, the system performs the retrieval and shows the results in a list-based widget. Their main drawback [3] is the difficult for users to memorise and master filtering and operators syntax (e.g., "and", "or" operator, etc.). *Exploratory search* [3] considers multiple iterations involving learning and investigation activities with higher-level goals (e.g., comparison, analysis, synthesis and evaluation). *Exploratory Search Systems* [2] aim to involve and engage users actively into the search process by providing human control over the seeking process. They aim to provide features to (re)formulate queries, giving information on the search space and clues for further possible search directions [4], allowing the constant exploration and filtering of retrieved results [3]. The presentation of query results is imperative to engage the user in the search process [5].

One popular technique to help users in deciding what to do next is the grouping of results [6]. Two main approaches exist: clusters and facets. The clustering approach [7] groups, often automatically, the query results according to similarity metrics. In the faceted approach, meaningful items' feature types are identified in advance, mostly manually, and labelled. Thus, *facets* are categories to characterise items in a collection. Each item is automatically enriched with multiple facets' labels. The query results can be grouped together based on these labels which form a categorisation, and can be used to further explore the space of results. According to White and Roth [8], exploratory search tools should "support querying and rapid query refinement" and "offer facets and metadata-based result filtering". The use of clusters or facets aids searchers with the query formulation that "significantly improves results" [5]. Flamenco [6] is a web-based system where the navigation is performed through the selection of hyper-links containing facets' labels. Relation Browser [9] is another example, it has two views, a list of facets and a cloud of facets labels; the user can filter result by choosing the facets. The system mSpace [10] uses a multi-column faceted browser for multimedia data. Carrot2 [11] is a web search engine that supports the navigation of web results through the selection of cluster hyper-links showed in a tree-based widget alongside of the list of results. In a testing with real users using Carrot2 [5], the clusters were useful in providing other relevant keywords to narrow the search and for serendipity search. Other systems that relies on the clusters of results have been introduces, like Vivisimo and SnakeT [12]. AcquaBrowser Library [13] does not show the cluster names in a listed way, but introduces a fluid, attractive and interactive word cloud visualisation, clicking on a word the user can refine his/her search. With the introduction of touch mobile devices, classical list-based result presentation poses challenges for the interaction with Exploratory Search Systems to refine searches [4].

From the GUI point of view, exploration systems are now exploiting visualisations to graphically display groups of items to provide an initial overview, permitting interactions to formulate queries and update the visualisation. This interaction "*overview first, zoom and filter, then details-on-demand*" is known as information seeking mantra [14]. Interactions can be grouped into: overview, navigation, and manipulation operations [15], and optionally, based on the application domain, interactions to compare the results of the query. FacetMap [16] is based on the facet concept and exploits a 2D visualisation to support dataset exploration. Facets are represented by ellipses and navigation is performed by clicking on the ellipses. ExploraTool presented in this paper is also based on the facets concept, but in order to scale on the number of facets, it organises them hierarchically. As example of a manipulation operation, the "*hierarchy manipulation*" term refers to a set of interactive operations performed on a hierarchical visualisation to directly and interactively change, re-order, move or copy its items; for example re-ordering through the drag-and-drop of hierarchical items. Lutz et al. [15] described many types of hierarchy manipulations by diagrammatically depicting their use and effects. ExploraTool has a hierarchical manipulation operation to change the order of the visualised facets.

In the visualisation field, many alternatives have been proposed to overcome the limits of classical list-based widgets of items, due to the impossibility of showing all items of a large dataset. One popular approach is the 2D space-filling visualisation technique that aims to exploit all of the available screen-space to show the dataset content. This technique divides the available screen space recursively using a basic shape (e.g., rectangle, circle). In this way parent-child relationships are represented as nested shapes, and sibling nodes are represented as neighbouring shapes at same depth. The most popular 2D space-filling visualisation is Treemap, introduced by Shneiderman during 1990 to provide a compact file system visualisation and to be able to identify at a glance the directories that take up the most of the space on the hard drive. Treemap has been extensively used to visualise intrinsically hierarchical data [17], [18], providing an overview of an entire dataset at a glance. Ellimap [19] is another type of 2D space-filling visualisation approach, which uses ellipses instead of rectangles to represent the nodes. Usually, shapes of 2D space filling visualisations are area proportional to a given metric (e.g., the number of items), visually giving an overview on this value. ExploraTool, presented in this paper, is a visual exploration system that exploits the use of ellimap layout with additional interaction functionalities (e.g., drill-down, roll-up), and provides dataset overview through the visualisation of facets, which are organised hierarchically on multiple levels. In this way, ExploraTool supports the constant exploration and filtering through selection of facets [8].

ExploraTool needs to interoperate with multiple repositories that can store data in different formats. In the CFD field most simulator software strategically use closed and proprietary data formats to make it expensive for customers to change their software products [20], [21]. ExploraTool relies on the plug-in based Floasys Architecture [22], [23] to be able to com-



SELECTION DETAILS

Value: *BrandD*

Items: *1147*

Current Path: *simulations (2248)*

Simulation Group	Num. of sims
> BrandD > Seg. B > ModelU > Diesel > 1,3_Multijet_16V > Rev. 1	1
> BrandD > Seg. B > ModelU > Diesel > 1,3_Multijet_16V > Rev. 15	1
> BrandD > Seg. B	535

Fig. 1. ExploraTool's Graphical User Interface (GUI). The central overview shows the simulation data set grouped together via their relevant attributes: *brands, segments, models, power sources, engines, and revisions*. The ellipses depict groups of simulations and the prioritised order of the attributes is indicated in the hierarchy column on the left. The user can filter the data set by clicking on any group of simulations (drill-down). If the user desires to return to the previous view, he/she can click on the container ellipse (roll-up). The navigation bar (underneath the main view) shows the path followed during the exploration; in the example, the path is the universe overview, labelled as "*Current Path: simulations*". If user desires a comparison of the performances of one group he/she can click on the "+ *Add*" button which adds the current group to the table below and the chart automatically updates accordingly.

municate with many software and data sources. Floasys has an API interface which acts as isolation layer, and a common data format to communicate with external client applications. In order to perform data integration he2014integration the API interface is implemented by other specifically designed software modules which read data from closed and open sources. Floasys architecture processes data from interoperable standards and protocols like CFD General Notation System (CGNS), and can be extended to other interoperable standards.

III. ExploraTool Features

This Section introduces the ExploraTool's features: dataset overview, exploration and vehicles' performances comparisons. The user task is to select a single simulation, or groups of simulations, in order to compare the vehicles' performances (e.g., aerodynamic performances) via an appropriate chart.

A. Data set Overview

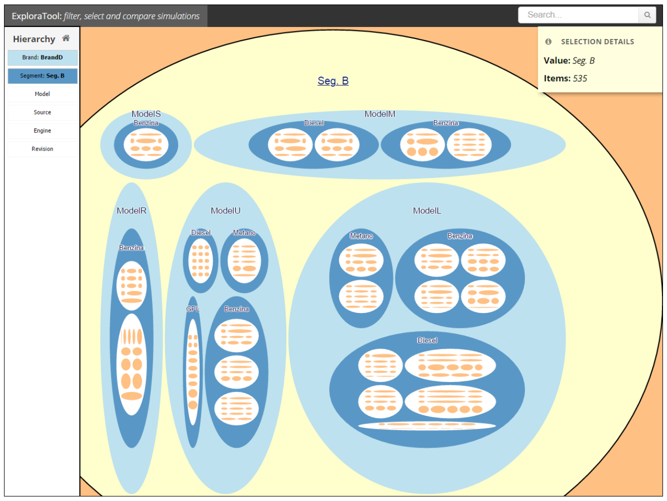
The ExploraTool's GUI (Fig. 1) has a central view which graphically depicts the simulations available within the repository, and a chart on the bottom to compare the selected vehicles' performances (e.g., aerodynamic performances). ExploraTool starts with an initial overview of the data set where the items are progressively grouped together by their main relevant attributes. The screenshot shows the simulations progressively grouped by Brands, Segments, Models, Power Sources, Engines, and Revisions. Brands and model names shown in Fig. 1 and throughout the paper have been anonymised, replacing them with artificial names. The hierarchy on the left shows the default initial ordering of attributes, based on the feedback provided by analysts in a large automotive manufacturer [22], but it can be changed any time by the user. Additional details on each group are requested by hovering the mouse cursor over the corresponding ellipse, causing additional information to be presented in a yellow tool tip box (see the top-right of Fig. 1). The same tool tip box could be used to show additional statistical information (such as the average, minimum and maximum drag-coefficient Cx values), as suggested by users during the Usability Study (Section V). ExploraTool tries to exploit all of the available screen space using ellimap [19]. Each group of simulations is a set depicted by an ellipse. In Fig. 1, the external white space represents the universe of all simulations within the repository. This is divided into subsets, depicted as ellipses, generating a nested-ellipse layout. For instance, the ellipse labelled *"BrandD"* in Fig. 1 represents the set of all simulations enriched with facet BrandD. The ellipse labelled *"Seg. A"* nested in the ellipse *"BrandD"* represents the set of all simulations of Seg. A with BrandD. In order to reduce cluttering, the tool shows only labels for two levels at a time. Each ellipse's area is chosen to be proportional to the number of simulations in that group, so that a user can obtain a quick perceptual overview of the spread of simulations. Alternatively, one could assign other measures to the area, such as the vehicles' performances for that group.

TABLE I
TYPICAL EXPLORATOOL TASKS.

Task #	Example
Task 1	*Selection of a Group* BrandF, Seg. C, ModelY
Task 2	*Selection of a Case* BrandB, Seg. C, ModelB, Fuel Petrol, 1.8TBI 16V, Rev. 5
Task 3	*Comparison Case vs. Group* *Case*: BrandD, Seg. B, ModelU, Diesel, 1.3 Multijet 16V, Rev. 2 *Group*: BrandD, Seg. B
Task 4	*Comparison Case vs. Case* *Case 1*: BrandB, Seg. C, ModelB, Petrol, 1.8 TBI 16V, Rev. 3 *Case 2*: BrandB, Seg. C, ModelB, Petrol, 1.8 TBI 16V, Rev. 6 *Case 3*: BrandB, Seg. C, ModelB, Petrol, 1.8 TBI 16V, Rev. 8
Task 5	*Comparison Group vs. Group* *Group 1*: BrandD, Seg. B *Group 2*: BrandF, Seg. B

B. Data set Exploration

The users can interactively explore the data set through an *in-depth navigation* [24] performing drill-down and roll-up operations. **Drill-down** occurs when a user has identified a potentially interesting group of simulations and he/she wishes to further explore the group, so he/she can click on the ellipse to obtain more details. ExploraTool shows multiple nested ellipses, so the user can drill-down one level at time or multiple-levels in one step by clicking on the internal nested ellipses (Fig. 2). Every time the user drills down in the hierarchy, he/she is effectively performing a refinement of the query, filtering all of the simulations in the repository. For instance, when the user selects in sequence the ellipses $BrandF$ -> $Seg.\ C$ -> $ModelY$ (Task 1 in Table I), he/she is performing a query to retrieve from the repository all the items with exactly these values (ellipse labels). For each click, ExploraTool smoothly enlarges the selected group, rendering a fast transition to the new view. **Roll-up** operation is performed when the user wants to have a global data set view, he/she traverses up the hierarchy by clicking on the container ellipse.


ExploraTool: *filter, select and compare simulations*

SELECTION DETAILS
Value: *Seg. B*
Items: *535*

Fig. 2. This effect of the drill-down operation by directly clicking on the ellipse with the label "Seg. B" from the main view of Fig. 1 is shown. The ExploraTool smoothly enlarges the selected group, rendering a fast transition between views. If the user desires to go back and see less details, he/she can click directly on the external space to perform a roll-up operation returning back by one level at a time. In order to return to the initial overview, as in Fig. 1, the user can click on the "Home" icon shown on the top-left.

As shown in Fig. 1, the tool shows a vertical navigation hierarchy bar on the left, which: (1) gives an overview of the hierarchy [14]; (2) shows the current depth during the simulation repository navigation, supporting user orientation; (3) permits the hierarchy re-arrangement by interactively swapping the facet's levels, showing visual cues to indicate for instance what happens if the mouse is released [15]. In addition, during interactive data set exploration, ExploraTool shows the navigation path in the *navigation bar*, indicating the total number of filtered items, and updates the hierarchy bar (displaying only the remainder of the hierarchy from the current position). Fig. 1 shows the whole universe and the total number of simulations (2248) within the repository.

The operations provided by the ExploraTool rely on the direct manipulation [15] principle, which concerns the direct interaction and manipulation of the rendered objects (e.g., directly clicking on the target object). The use of ellipses as basic shapes guarantees there will be space between sibling ellipses at same level and amongst nested ellipses. This extra space improves the hierarchy discernment [19] and every operation involves exactly the target shape. For instance, in order to drill down in the hierarchy, the user points and clicks on exactly the nested ellipse required. In order to roll-up the user points and clicks on exactly the parent shape utilising the space between the parent and child ellipses, which is always present. In other 2D space-filling techniques, such as the Treemap visualisation technique, both nested rectangles and adjacent rectangles have no space between them, so *the position of the mouse pointer designates a branch of the tree* [24] because *each point belongs to a single leaf node but also to all its ancestors* [24].

C. Vehicles' Performance Comparison

When the user finds an interesting data set he/she can add it to basket of simulations to compare by clicking on the "+ Add" button (Fig. 1). The user explores the repository and adds groups of items to the comparison bar (bottom of Fig. 1). Every time a selection is added to the basket, ExploraTool updates the chart on the right, showing the mean, maximum and minimum value for each selection, facilitating the comparison. Hovering the mouse cursor on the chart displays the exact values from the chart.

IV. Algorithm to process Dataset

Interactive exploration through ExploraTool is performed by clicking on ellipses (drill-down and roll-up operations). Nested ellipses form a hierarchy (tree data structure). Every time the user drills down, he/she clicks on ellipse that identifies a branch in the hierarchy. This Section introduces an algorithm called `BuildHierarchy` to process any tabular dataset to generate one of the possible tree data structures to be visualised with any kind of 2D space-filling visualisation techniques (e.g., Treemap, ellimap, etc.). The algorithm has been used within ExploraTool to process data from vehicle simulation repositories and organise the identified facet categories (i.e., *brand, project model, power source, engine type*) in an initial hierarchical ordering (decided in advance by the analysts) but can be altered any time through a re-arrangement operation that triggers the hierarchy recomputing. The algorithm can process any other dataset to be explored through ExploraTool, such as a catalogue of parts. In order to process other datasets, the requirement is to identify the facets (attributes) of the items and their values. The `BuildHierarchy` algorithm is implemented in the back-end of ExploraTool, which makes use of the generated tree data structure to build the ellipse-based visualisation. It runs in three cases: (1) when the user opens ExploraTool for the first time, the algorithm builds the initial default partial tree made of the first r levels; (2) every time the user performs a drill-down operation, he/she is navigating to a specific branch of the entire tree and so the algorithm builds a new subtree with r levels rooted at the selected node (the server provides chunks of subtrees made by r levels); (3) every time the user performs the re-arrangement operation, by sorting the facets in a different order.

A. BuildHierarchy Algorithm Description

Fig. 3 shows an abstract example of the `BuildHierarchy` algorithm (Alg. 1). It takes a dataset as input, and to be independent of the specific technology to store simulations, it has been transformed as a tabular dataset where each row records the relevant metadata for an individual simulation and the columns are their attributes. So, the **input** is a collection of n items $R = \{s_1, ..., s_n\}$. The dataset R (left side of Fig. 3) has a row for each individual item $s \in R$ (i.e., a simulation) that is described through attributes attached to it. The set $A = \{a_1, a_2, ..., a_m\}$ contains the labels/names of the attributes used to describe the items. For example, in the simulations use case, the labels for the facets [16] are $A = \{brand, projectmodel, powersource, enginetype\}$. The dataset has these facets (attributes) on the columns.

The **algorithm output** is a tree data structure $T = (V, E)$ (right side of Fig. 3), where V is the set of nodes and E are the edges. For each attribute $a_i \in A$, there will be a level in the tree T (the height of tree is exactly the number of facets). Each attribute a_i has a set of valid values called domain $D_i = dom(a_i)$. For instance, in the simulation context, the attribute *power source* has the domain $D_{powersource} = dom(powersource) = \{Bifuel, Petrol, Diesel\}$. In the example, the attribute *brand* has the following valid values $D_{brand} = dom(brand) = \{BrandA, BrandB, BrandC, BrandD, BrandF\}$. At the level of the tree corresponding to attribute a_i, there are the nodes with the values in D_i. Some nodes corresponds to zero items in the original data set, so they are not present in the hierarchical view. For instance, the path $(root, 1, B, b)$ that would be present in the full tree is not present in our constructed tree because there are no items in the original data set with the values $s[1] = 1 \wedge s[2] = B \wedge s[3] = b$.

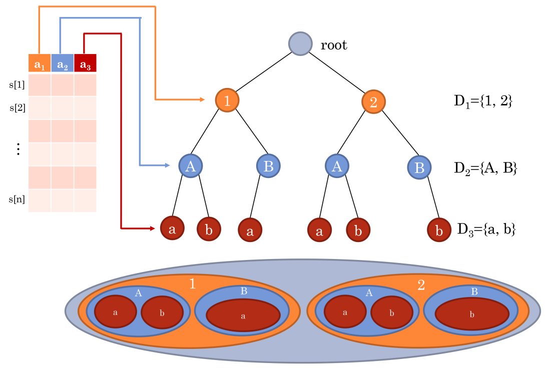


Fig. 3. Example of hierarchy extraction from a table. The table on the left has tree attributes $A = \{a_1, a_2, a_3\}$ and n simulations. The tree on the right has a level for each domain D_i and at each level i there are the nodes with the labels in D_i. ExploraTool builds the visualisation shown on the bottom of the figure, starting from the tree data structure.

The tree T will be displayed as nested ellipses starting from the root (Fig. 3 bottom side). For each node, the algorithm stores the label to be displayed on the ellipses, and calculates the number of simulations ($Count[u] \in \mathbb{N}, \forall u \in V$) that will determine the area of the relevant ellipse.

Algorithm 1 shows the algorithm pseudo-code to process the dataset as input and generate the tree data structure. Initially the tree T has the *root* node and no edges ($E = \emptyset$). The algorithm scans all the simulations in the repository $s \in R$ exactly one time (Alg. 1 line 5). For each simulation s, the algorithm scans the simulations' attributes in the order specified by the function f_S (Alg. 1 line 6). So, the algorithm scans the dataset row by row, and for each row the columns.

Algorithm 1 takes each simulation and traverses down the tree from the root to a leaf, one level at time, through the nodes with simulation attribute values. On line 7, the algorithm tries to find the node at level i with value $attrvalue$. If the node does not exist, the algorithm creates it at lines 8-11. At line 12, the algorithm counts the number of simulations; this value will be used to calculate the ellipse's areas.

Algorithm 1: BuildHierarchy(R, r, f_S)

```
Input: Set of simulations $R$, number of levels $r$ to load and an
       attributes ordering function $f_S$
Output: $T = (V, E)$
1  $root \leftarrow$ create the hierarchy root;
2  $V \leftarrow \{root\}, E \leftarrow \emptyset$;
3  $prevnode \leftarrow root$;
4  $curnode \leftarrow null$;
5  foreach simulation s in the repository R ($s \in R$) do
6      foreach attribute value "attrval" in s,
7      ordered by $f_S$ and limited to the first $r$ values do
8          $curnode \leftarrow$ Find the child with value $attrval$;
9          if curnode is null then
10             $curnode \leftarrow$ create a new node;
11             $Label[curnode] \leftarrow attrval$;
12             $Count[curnode] \leftarrow 0$;
13             $V = V \cup curnode$;
14             $E = E \cup \{(prevnode, curnode)\}$;
15         $Count[curnode]++$;
16         $prevnode \leftarrow curnode$;
17     $prevnode \leftarrow root$;
18 return $(V, E)$;
```

The f_S function in input specifies a sorting of the facets (sorting of domains $f_S : \{1, 2, \cdots, m\} \rightarrow \{1, 2, \cdots, m\}$), which has a direct impact upon the resulting hierarchy. The algorithm creates an initial hierarchy based on an initial domain sorting function f_S specified at configuration time. Then, the tree is rendered using a 2D space-filling visualisation approach. The user can swap the facets through a rearrangement operation by dragging levels in the hierarchy, choosing a different permutation of the facets that changes the input function f_S to obtain a different hierarchical views.

B. BuildHierarchy Algorithm Running time discussion

The algorithm generates a tree data structure with height r, where each level is a facet (attribute). At each level the tree has a node for each attribute value. For instance, in the hierarchy at the level of the segment attribute, there is a node for each segment value (e.g., Seg. A, Seg. B, Seg. C, etc.). In the worst case all attributes have exactly p values (p nodes at each tree level). Therefore, in the worst case the total number of nodes in the tree is $O(p^r)$ with $p > 1$ and $1 \leq r \leq m$. Introducing the parameter r, two interesting features can be provided: (1) details on-demand when the user drills-down in the hierarchy, it is asking for the next r levels; (2) from a computational point of view, the algorithm computes only r levels at time. In order to keep the hierarchy clear during the visualisation, ExploraTool loads five facets at time; it does not show more than five levels at each time ($r \leq 5$), so the number of the nodes in the tree is in the worst case $O(p^5)$.

In order to build the tree data structure, the algorithm scans all the dataset items ($\Theta(n)$ where $|R| = n$) and exactly r columns ($O(r)$, the chosen facets to visualise). Columns are scanned in the order defined by the sorting function f_S, which is chosen by the user via the GUI. Given the value in the intersection of the selected row (simulation) and column (attribute), the algorithm checks whether a node with that value is in the tree at level r. Thus, the algorithm `BuildHierarchy` **running time is**: $\Theta(n) \cdot O(r) \cdot O(p) \leq O(n \cdot r \cdot p)$.

Furthermore, the algorithm to process any tabular dataset and build a hierarchy can be executed by using the map/reduce paradigm on large data sets in a distributed environment using for instance Apache Spark[1]. The idea is to slice the tabular dataset in q groups of rows. Each group of row will be processed by a computing node, executing the algorithm and generating a tree data structure. The trees generated by all computing nodes can be merged together to obtain the final hierarchy displayed by ExploraTool.

V. User Evaluation Study

ExploraTool adopts a novel visual and interactive user interface for the engineering field to explore multiple repositories of simulations. This section reports on the fundamental activity to evaluate the tool's effectiveness and usability in a real setting involving industrial experts from a large automotive manufacturer. The question to answer is how users perceive the system, evaluating its effectiveness, in terms of tasks completion as well as the usability of the interface and the overall satisfaction, acquiring user feedback.

A. Methodology

The study lasted thirty minutes for each participant and consisted of four phases as described in the following.

1) Preliminary Survey: The participant fills out a questionnaire[2] to collect demographic information, particularly as related to their experience in the simulation field and their existing procedures to compare vehicle performances using simulation repositories. ExploraTool uses colours in its user interface, so it is fundamental to examine its usability by people with colour deficiencies, especially the effectiveness of the chosen colour blindness colour palette. Therefore, a quick Colour Blindness test has been performed (Ishihara test).

2) Training Phase: It demonstrates the functionality to the participant, using *training material* with standard basic tasks to ensure consistency of the explanation among the participants.

3) Testing Phase: Users execute five tasks (Table I) using ExploraTool in which he/she should find and select single simulations and/or groups of simulations to compare their performances. At the end of each task, the user answers questions to evaluate whether it was successfully completed,

[1]Apache Spark web-site http://spark.apache.org/

[2]ExploraTool Usability Study Questionnaires are available on-line at http://floasysorg.github.io/Floasys/usabilitystudy/exptoolv1.html

rate how ease and quick it was to perform the task (standard questions from the After Scenario Questionnaire[3]).

4) Summary Survey: The questionnaire concludes with the assessment of the overall ExploraTool perceived usefulness and user satisfaction. This part of the questionnaire is based on a standard TAM model, which is extensively used in usability studies to explain and/or predict users' behavioural intentions when accessing a new technology or system as well as to test the user acceptance.

B. Evaluation Results Analysis and Discussion

1) Participants' Demographics: Recruits comprised 24 engineers of a large automotive manufacturer. The sample was mostly male (87.5%) with mean age of 34 (std. dev. 7.5). The test has been performed in two locations in Italy: one day in Pomigliano D'Arco (Naples) involving 8 participants and almost two days in Orbassano (Torino) involving 16 participants. The conditions were kept the same (same hardware and software) in both locations, using an isolated room containing only one participant at time, where he/she could concentrate on the test without distractions.

ExploraTool has been designed primarily for simulation Analysts but different company roles performed the usability test, all of whom involved in the CFD simulation field: 16 CFD Analysts, 3 Performance Engineers (PEs) and 5 Technical Managers (TMs) with, on average, 4, 8, and 12 years of experience in the field, respectively. Technical Managers and Performance Engineers usually perform few simulations per year (mean 85, std. dev. 69) as compared to the analysts (mean 151, std. dev. 123), because TMs are responsible for the internal team organisation, resource monitoring and their allocations, whilst PEs work on big picture projects and are responsible for design choices. There was only one colour blind participant; whilst he was not able to distinguish colours, he successfully completed all the tasks, identifying and discerning correctly the ellipses. ExploraTool uses a space filling visualisation approach using ellipses to depict groups of simulations. As such, it is important to understand which users had previous experience with such visualisations. Participants had high experience with standard charts (bar, pie, chart and surface charts) used for the everyday work, but much less had experience with Treemap (4%).

For all participants it would be useful in their role to be able to automatically extract simulation data from the repositories and to obtain comparisons of related statistics among different releases of the same project or different projects releases (agreement of 100%). Notwithstanding, all participants declared that they do not have an automatic tool to perform these tasks. Instead the common procedure is to export data from the simulator software in comma separated value format and to analyse them via Microsoft Excel.

2) Tasks Execution Results: The users executed tasks corresponding to those in Table I. In order to assess the effectiveness of the tool the error rate has been measured for each task. As a result, all participants completed tasks 1, 2, 4, and 5 without errors, whilst task 3 had an error rate of 2.8%. A simulation analyst wrongly selected an alternative group of simulations instead of the group $BrandD$ -> $Seg.B$ requested in Task 3 (Table I). The responses to the ASQ questionnaire indicated that participants were highly satisfied with the ease of the tasks and the amount of time required to complete them; across all of the tasks there was a mean of 6.9 (std. dev. 0.1) for the easiness and 6.7 (std. dev. 0.1) for the time, on a 7-point scale. Furthermore, grouping the participants by their main role (CFD Analyst, Technical Manager, Performance Engineer) in the company yields no statistical difference with regard to the above metrics (Kruskal-Wallis test).

3) Perceived Usefulness and overall Acceptance Results: At the end of the Testing Phase participants responded to the TAM questionnaire, whose Cronbach's Alpha value was 0.92.

TABLE II
SPEARMAN'S CORRELATION COEFFICIENTS BETWEEN SUBSCALES: PU, PERCEIVED USEFULNESS; EOU, PERCEIVED EASE OF USE; ATT, ATTITUDE TOWARD USE; BI, BEHAVIOURAL INTENTION TO USE.

Subscale	PU	EOU	ATT	BI
PU	1.0			
EOU	.194	1.0		
ATT	$.604^{**}$	$.457^{*}$	1.0	
BI	$.530^{**}$	.384	$.557^{**}$	1.0

$^{**}p < .01$, $^{*}p < .05$

Table II reports the Spearman's correlation coefficients among the subscales with the corresponding significance levels (indicated by the $*$ and the p value). In the table the highest correlation is between ATT and PU (.604, with p value $< .01$). BI is positively correlated with PU and ATT with high significant level ($p < .01$ for both metrics). Furthermore, analysing the TAM answers' rates (7-point Likert scale) on the PU, EOU, ATT, BI subscales, results were highly positive for all these metrics. The highest rate was for the question "*Using the system would enable me to accomplish tasks more quickly*" in the Perceived Usefulness questionnaire section (mean 6.8, std. dev. 0.4). In addition, a regression analysis was carried out in order to identify which variables (PU, EOU, and ATT) influenced the use of ExploraTool (BI). The model yielded an adjusted R^2 value of .606. Based on the analysis of the attitude results, participants think that ExploraTool's idea is wise, smart and interesting. As result, ATT is a significant variable in increasing the software's acceptance. When asked to express the positive tool aspects, the participants indicated: easiness (71%), quickness (58%), intuitiveness (25%), usefulness (17%) and effectiveness (13%). The negative aspects concerned mainly the visualisation (29%), in particular the partial overlapping of some labels and some thin ellipses, pointing out the need for improvements of the visualisation overall aesthetic. In addition users reported future tool improvements, like additional aggregated statistical data on mouse hovering (13%) and search by keywords (4%). In summary, despite the participants' lack of knowledge of space filling visualisations like Treemap, they were able to complete the tasks, and expressed high satisfaction in terms of its usefulness, usability, and simplicity. Furthermore, users would use the tool on regular basis and recommend other to use it (questions D18 and D19 of the questionnaire).

[3]ASQ Questionnaire http://garyperlman.com/quest/quest.cgi?form=ASQ

VI. Conclusions and Future Work

This paper presented ExploraTool to visually and interactively explore multiple repositories of simulations through a novel user interface for the engineering field. As practical industrial application, the tool has been used to select individual and/or groups of simulations to compare their performances (i.e., comparison of aerodynamic simulations by the drag-coefficient, called Cx). The utility of ExploraTool has been evaluated in a real setting with expert industrial users. The users found the tool useful, usable and simple to use, and users were interested in the novel ExploraTool interactive user interface. The ExploraTool concept generalises beyond the simulation context to any other context in which the goal is to select items by their properties and perform comparisons. Filling out the questionnaires, analysts indicated the need to select and compare not only simulations, but also wind tunnel experiments from a cleaned repository and competitors' products performance data, extending the applicability of the tool. As possible extension to other industries, the tool can provide a visual overview of catalogues of parts, supporting the interactive finding of parts by their properties.

Reflections on the process adopted to design ExploraTool may serve to provide good practice suggestions for the design of novel interactive user interfaces for the engineering field. In order to identify appropriate tasks and type of interactions, an essential element was the close interaction with the end-users, adopting an agile development methodology consisting of two week periods for the plan, design, develop, and internal testing phases. For ExploraTool, the crucial stimulation of discussions and user engagement was a small proof of concept tool permitting the basic interactions. In this case the users were willing to pro-actively participate in discussions and support its development.

As future work, users already provided interesting requests for new features during the evaluation test: the facility to bookmark the exploration of results; to introduce a specific preference section; to filter the repository items by typing a keyword within a search bar that updates the visualisation with the filtered items; the ability to export the comparisons in Excel and PowerPoint in a manner which is compliant to the internal industrial templates. In addition, improvements to the layout algorithm are needed to avoid thin ellipses, thereby improving the overall visualisation aesthetic, whilst also improving the label positions.

Acknowledgment

The authors gratefully thank all participants who took part to the ExploraTool Usability Study.

References

[1] D. A. Keim, "Visual exploration of large data sets," *Communications of the ACM*, vol. 44, no. 8, pp. 38–44, 2001.

[2] G. Marchionini, "Exploratory search: from finding to understanding," *Communications of the ACM*, vol. 49, no. 4, pp. 41–46, 2006.

[3] E. F. Duarte, E. Oliveira Jr, F. R. Côgo, and R. Pereira, "Dico: A conceptual model to support the design and evaluation of advanced search features for exploratory search," in *Human-Computer Interaction–INTERACT 2015*. Springer, 2015, pp. 87–104.

[4] K. Klouche, T. Ruotsalo, D. Cabral, S. Andolina, A. Bellucci, and G. Jacucci, "Designing for exploratory search on touch devices," in *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems*. ACM, 2015, pp. 4189–4198.

[5] M. Burt and C. Li Liew, "Searching with clustering: An investigation into the effects on users' search experience and satisfaction," *Online Information Review*, vol. 36, no. 2, pp. 278–298, 2012.

[6] M. Hearst, "Clustering versus faceted categories for information exploration," *Communications of the ACM*, vol. 49, no. 4, pp. 59–61, 2006.

[7] A. L. Kaczmarek, "Interactive query expansion with the use of clustering-by-directions algorithm," *IEEE Transactions on Industrial Electronics*, vol. 58, no. 8, pp. 3168–3173, 2011.

[8] R. W. White and R. A. Roth, "Exploratory search: Beyond the query-response paradigm," *Synthesis Lectures on Information Concepts, Retrieval, and Services*, vol. 1, no. 1, pp. 1–98, 2009.

[9] R. G. Capra and G. Marchionini, "The relation browser tool for faceted exploratory search," in *Proceedings of the 8th ACM/IEEE-CS joint conference on Digital libraries*. ACM, 2008, pp. 420–420.

[10] M. Wilson, A. Russell, D. A. Smith *et al.*, "mspace: improving information access to multimedia domains with multimodal exploratory search," *Communications of the ACM*, vol. 49, no. 4, pp. 47–49, 2006.

[11] C. Carpineto, S. Osiński, G. Romano, and D. Weiss, "A survey of web clustering engines," *ACM Computing Surveys (CSUR)*, vol. 41, no. 3, p. 17, 2009.

[12] P. Ferragina and A. Gulli, "A personalized search engine based on web-snippet hierarchical clustering," *Software: Practice and Experience*, vol. 38, no. 2, pp. 189–225, 2008.

[13] J. Kaizer and A. Hodge, "Aquabrowser library: search, discover, refine," *Library Hi Tech News*, vol. 22, no. 10, pp. 9–12, 2005.

[14] B. Shneiderman, "The eyes have it: A task by data type taxonomy for information visualizations," in *IEEE Symposium on Visual Languages,*, 1996, pp. 336–343.

[15] R. Lutz, D. Rausch, F. Beck, and S. Diehl, "Get your directories right: From hierarchy visualization to hierarchy manipulation," ser. VL/HCC, Melbourne, VIC, 2014, pp. 25–32.

[16] G. Smith, M. Czerwinski, B. R. Meyers, G. Robertson, and D. Tan, "FacetMap: A scalable search and browse visualization," *IEEE Transactions on Visualization and Computer Graphics,*, vol. 12, no. 5, pp. 797–804, 2006.

[17] P. Demian and R. Fruchter, "Finding and understanding reusable designs from large hierarchical repositories," *Information Visualization*, vol. 5, no. 1, pp. 28–46, 2006.

[18] J. Bauder and E. Lange, "Exploratory subject searching in library catalogs: Reclaiming the vision," *Information Technology and Libraries*, vol. 34, no. 2, pp. 92–102, 2015.

[19] B. Otjacques, M. Cornil, and F. Feltz, "Visualizing cooperative activities with ellimaps: the case of Wikipedia," in *Cooperative Design, Visualization, and Engineering*. Springer, 2009, pp. 44–51.

[20] C. Gargiulo, D. Pirozzi, and V. Scarano, "An architecture for CFD workflow management," in *Proceedings of the 11th IEEE International Conference on Industrial Informatics (INDIN), Bochum, Germany, July 29-31*, 2013, pp. 352–357.

[21] C. Gargiulo, D. Pirozzi, V. Scarano, and G. Valentino, "A platform to collaborate around CFD simulations," in *Proceedings of the 23rd IEEE International Conference on Enabling Technologies: Infrastructure for Collaborative Enterprises (WETICE)*, 2014, pp. 205–210.

[22] C. Gargiulo, D. Malandrino, D. Pirozzi, and V. Scarano, "Simulation data sharing to foster teamwork collaboration," *Scalable Computing: Practice and Experience*, vol. 15, no. 4, pp. 309–329, 2014.

[23] A. Fish, C. Gargiulo, D. Pirozzi, and V. Scarano, "Simulation repository visualisation and exploration," in *13th IEEE International Conference on Industrial Informatics (INDIN)*, 2015, pp. 832–837.

[24] R. Blanch and E. Lecolinet, "Browsing zoomable treemaps: structure-aware multi-scale navigation techniques," *IEEE Transactions on Visualization and Computer Graphics*, vol. 13, no. 6, pp. 1248–1253, 2007.