Egyptian Informatics Journal 23 (2022) 55–64

Contents lists available at ScienceDirect

Egyptian Informatics Journal

journal homepage: www.sciencedirect.com

Full length article

Retrieval of behavior trees using map-and-reduce technique

Safia Abbas [a,b], Rania Hodhod [c,*], Mohamed El-Sheikh [d]

[a] Computer Sciences Department, College of Computer and Information Sciences, Princess Nourah bint Abdulrahman University, Saudi Arabia
[b] Computer Science Department, Faculty of Computer and Information Sciences, Ain Shams University, Egypt
[c] TSYS School of Computer Science, Columbus State University, USA
[d] Basic Science Department, Cairo University, Egypt

A R T I C L E I N F O

Article history:
Received 22 December 2020
Revised 19 March 2021
Accepted 26 May 2021
Available online 11 June 2021

Keywords:
Information retrieval
Behavior trees
Cognitive scripts
Map-and-reduce
Pharaoh

A B S T R A C T

There has been an increased interest in the creation of AI social agents who possess complex behaviors that allow them to perform social interactions. Behavior trees provide a plan model execution that has been widely used to build complex behaviors for AI social agents. Behavior trees can be represented in the form of a memory structure known as cognitive scripts, which would allow them to evolve through further development over multiple exposure to repeated enactment of a particular behavior or similar ones. Behavior trees that share the same context will then be able to learn from each other resulting in new behavior trees with richer experience. The main challenge appears in the expensive cost of retrieving contextually similar behavior trees (scripts) from a repertoire of scripts to allow for that learning process to occur. This paper introduces a novel application of map-and-reduce technique to retrieve cognitive with low computational time and memory allocation. The paper focuses on the design of a corpus of cognitive scripts, as a knowledge engineering key challenge, and the application of map-and-reduce with semantic information to retrieve contextually similar cognitive scripts. The results are compared to other techniques used to retrieve cognitive scripts in the literature, such as Pharaoh which uses the least common parent (LCP) technique in its core. The results show that the map-and-reduce technique can be successfully used to retrieve cognitive scripts with high retrieval accuracy of 92.6%, in addition to being cost effective.

1. Introduction

Creating social intelligent agents (SIAs) that can behave in a human-like manner has been of interest to AI researchers. SIAs have played various roles in different domains including non-player characters (NPCs) in game environments [1], IoT assistive agents [2], narrative intelligence [3], and autonomous virtual agents or robots [4,5]. Representing and modeling complex AI behaviors for social agents has been classified as a cumbersome task for specialized researchers across multiple domains. Hence, different attempts have been done to model social behaviors including manually designed scripts, finite state machines (FSMs), machine learning techniques, and behavior trees (BTs) [6,7,8,9]. All these techniques depend on the targets and goals of the agents to model social behaviors [10,11,12]. Among these modeling techniques, BTs have been in the forefront and the most effective structured model [6] to capture and store the required information about the surrounding environment [13,2]. This can be attributed to the nature the social behaviors' structure which has the form of a series of events linked with preceding and succeeding ones, in addition to its flexibility to handle the internal relations between the connected events.

Many researches have been devoted to improve and develop the learning process in behavior trees using machine learning and graph theory [14] to either learn from observations [9] or learn from experience [8]. The main assumption in these works is the presence of the BTs of interest; there was no need to search for a similar BT. However, this is not usually the case as there might be situations where the SIAs need to look for similar BTs to learn from them to handle the current situation. Representing BTs as

* Corresponding author.

E-mail addresses: samahmoud@pnu.edu.sa (S. Abbas), hodhod_rania@columbusstate.edu (R. Hodhod), mohamedelsheikh@cu.edu.eg (M. El-Sheikh).

Peer review under responsibility of Faculty of Computers and Information, Cairo University.

Production and hosting by Elsevier

https://doi.org/10.1016/j.eij.2021.05.005
1110-8665/

cognitive scripts can allow for that learning process to occur by SIAs. Cognitive scripts allow for the change and further development over multiple exposure to repeated enactment of a particular behavior or similar ones. Retrieval of cognitive scripts has always been challenging because of the context and dependency of events [15,16]; some events can only occur if other events occurred prior to them, this is defined by the prerequisite relation. Moreover, defining the context of an event is another challenge since an event occurring in the beginning of a script might have a different meaning from when it occurred at the end of a script [17].

This paper introduces a new retrieval method for BTs represented in the form of cognitive scripts. The implemented method uses the map-and-reduce technique to retrieve contextually similar scripts to a current script (target) from a corpus of scripts. Contextual information of scripts can be captured via different relations, such as pre-requisite, is-a, and part-of relations provided by ConceptNet, a giant semantic network hosted on the cloud [18]. Pharaoh [17] was also implemented to compare its performance to the map-and-reduce technique in terms of run time. The rest of the paper is organized as follows: Section 2 presents the related work. Section 3 shows the building blocks, development, and the implementation of the corpus of cognitive scripts. Section 4 illustrates the map-and-reduce retrieval method followed by a practical example. Section 5 discusses the experimental results. Finally, Section 6 provides concluding remarks and future work.

2. Related work

Recently, information retrieval prominence from semantic, unstructured heterogeneous data that is delivered by multiple sources of electronic media has attracted many researchers, especially in the AI domain [19], and has been considered as one of the key challenges when developing assistive technologies [2,12]. Despite many researchers have been devoted to address the prediction problem from BTs using machine learning techniques like behavioral decision tree, rule-based machine learning, classification technique, and reinforcement learning behavior, they all focused on easy and small-sized BTs that allows for limited and fixed pre-determined future actions for a specific application or game scene. None of these research works has considered methods to allow learning between BTs, independent from a particular domain or tree size, by retrieving and using similar BTs to predict new unforeseen actions without any limitation to a particular scene/application or the existence of pre-canned expected actions [20,21,22]. Accordingly, new existing works have shifted the focus to the use of sensors to capture user behaviors and modeling them using BTs, in addition to working out ways to optimize and refine these BTs to allow for more logical trees for future learning goals. To the extent of our knowledge, no work has been done in the area of retrieving contextually similar BTs with the goal of allowing the evolvement of a BT of interest. The representation of BTs as cognitive scripts facilitates this retrieval process and allows for cross-learning to take place between the BTs.

Few works have been done to retrieve the most relevant cognitive scripts to a target script, most of which rely on exact matching between events that occur in the same levels in both the retrieved and target scripts [19,23,24]. This can be seen as a limitation as it might cause missing other scripts that can be similar enough but do not hold these two requirements. [17] presented Pharaoh, a contextually retrieval algorithm that relies on the least common parent (LCP) technique to find/define a common perquisite event for two events in two scripts; LCP helps define the context of scripts. Pharaoh retrieves the most similar cognitive scripts to a target script and organize them based on their relevance. The main drawback of Pharaoh is the high retrieval time, especially when it comes to large multi-branched scripts [17,21].

Recently, map-and-reduce technique has been prospered as one of the multiple techniques that have been developed and implemented to cope with the immense growth of multifarious types of data and excavate the pertinent information for further use in classification [25], scheduling [26,27], time and storage optimization [28,29], and clustering [30,31,32]. Despite the immense use of map-and-reduce technique in big data and information retrieval [13,14], it has not been considered as a way to look for and retrieve cognitive scripts from a corpus of cognitive scripts.

This paper presents a novel application of the map-and-reduce technique to retrieve similar cognitive scripts to a target script. The paper compares the performance of map-and-reduce with semantic knowledge to Pharaoh's performance. Map-and-reduce has provided higher accuracy rate over Pharaoh and very short retrieval time. The following section presents the structure of the corpus of cognitive scripts.

3. Corpus of cognitive scripts

The corpus of cognitive scripts consists of semi-structured NoSQL data in the form of JSON files and structured data in the form of relational tables - both the files and tables are managed by the database engine. In this section, the cognitive scripts' preliminaries, the semi-structured files and the internal relational tables are presented in detail, whereas the retrieval of scripts from the corpus is described later in Section 4.

3.1. Cognitive scripts preliminaries

Cognitive scripts are defined as hypothetical mental structures that describe everyday knowledge to reflect the understanding of events [33]. Cognitive scripts provide a representation of behavioral tasks in the form of connected events linked with preceding and succeeding ones [17]. In this paper, cognitive scripts are represented in predicate form where each node has an id and represents an event that is comprised of a verb, a subject and an object as follows: *verb (subject, object)*. Fig. 1 shows a sample cognitive script for attending a concert represented as a tree of nodes and edges where the nodes signify the events in predicate form and the connected edges show the temporal precedence of the events. Fig. 1 also shows how branches can occur in a cognitive script; node 303 can have a subsequent event node 307 or node 309. Table 1 provides a snippet of the structured table stored in the database with the predicate form of the tree nodes.

3.2. Semi-Structured files

This work represents BTs as cognitive scripts. The relation between the events in a cognitive script is retrieved from ConceptNet [18] and are then stored in JSON files. Fig. 2 shows the relation between the event 'sleep' being a prerequisite to the event 'dream', this contextual information is added to the edge connecting these two events in the following form: start node is "dream", relation is "HasPrerequisite", and end node is "sleep". The most significant fields are:

@id: unique field shows a short string of the start and end nodes, and the relation between them.

@type: represents the type, either an edge or a node.

weight: shows the edge strength in this assertion. If it is 1, the two words are identical.

end: shows the URL of the second argument of the assertion.

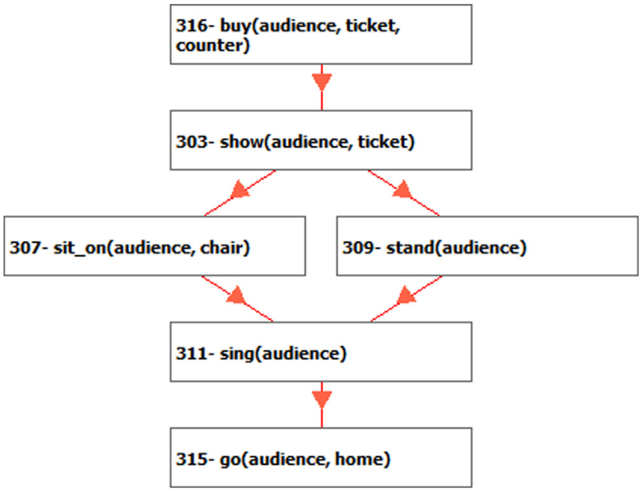

Fig. 1. Cognitive script for attending a concert [17]

Table 1
Attending a concert script represented in predicate form.

Parent node IID	Parent node Form	Child node IID	Child node Form
307	sit_on(audience, chair)	311	sing(audience)
309	stand(audience)	311	sing(audience)
303	show(audience, ticket)	307	sit_on(audience, chair)
303	show(audience, ticket)	309	stand(audience)
316	buy(audience, ticket, counter)	303	show(audience, ticket)
311	sing(audience)	315	go(audience, home)

dataset: represents the language of the ConceptNet, English language in our case.

surfaceText: a sentence expresses the meaning of an edge

start: shows the URL of the first argument of the assertion.

rel: shows the relation type of the assertion, Fig. 3 shows example of the relations.

The JSON files are stored locally as contextual information associated with the scripts. This helps to lessen the retrieval time when checking this information during the retrieval process. Moreover, the corpus has general contextual information for 2000 common words in the English language that were chosen from to www.talkenglish.com website and are stored in the form of txt files. In order to reduce the error cost and increase the agility of the script retrieval process, the corpus is designed to be scalable; new contextual information can be retrieved and stored automatically when the system receives new scripts that contain new words.

When a new script is received, the database engine crawls ConceptNet to find information about the new words in this script. For example, Fig. 4 shows a snapshot of the txt file for the new event "sleep". Each line in the txt file is a contextual piece of information for the term "sleep". For example, 'close eyes' is a pre-req to 'sleep' and 'dream' is a subsequent event. The "none" field means that the concept can be a noun, verb, or the concept consists of more than one word as in "get_in_bed". This information is then stored in the internal database and is used to define the relations between different events when needed. The crawling algorithm is described in Fig. 5.

3.3. Internal relational database

The internal database consists of multiple structured relational tables as shown in Fig. 6. It contains contextual information about the nodes (events) and the connecting edges in the cognitive script,

$\text{dream} \xrightarrow{\textbf{HasPrerequisite}} \text{sleep}$


```
"@id": "/a/[/r/HasPrerequisite/,/c/en/dream/,/c/en/sleep/]",
"@type": "Edge",
"dataset": "/d/conceptnet/4/en",
"surfaceText": "Something you need to do before you [[dream]] is [[sleep]]"
"weight": 7.745966692414834

"start": {
        "@id": "/c/en/dream",
        "@type": "Node",
        "label": "dream",
        "language": "en",
        "term": "/c/en/dream"
    },
"rel": {
        "@id": "/r/HasPrequisiteite",
        "@type": "Relation",
        "label": "HasPrerequisite"
    },
"end": {
        "@id": "/c/en/sleep",
        "@type": "Node",
        "label": "sleep",
        "language": "en",
        "term": "/c/en/sleep"
    },
```


Fig. 2. Prerequisite relations for "dream" word.

RelatedTo	HasSubevent	Synonym	SimilarTo
FormOf	HasFirstSubevent	Antonym	EtymologicallyRelatedTo
IsA	HasLastSubevent	DistinctFrom	EtymologicallyDerivedFrom
PartOf	HasPrerequisite	DerivedFrom	CausesDesire
HasA	HasProperty	SymbolOf	MadeOf
UsedFor	MotivatedByGoal	DefinedAs	ReceivesAction
CapableOf	ObstructedBy	MannerOf	ExternalURL
AtLocation	Desires	LocatedNear	
Causes	CreatedBy	HasContext	

Fig. 3. Types of relations used in ConceptNet knowledge base.

bed, RelatedTo, sleep, 12.5373, [bed] is related to [sleep], none, none
sleep, HasSubevent, dream, 9.797959, Something that might happen when you [sleep] is [you dream], none, none
sleep, HasPrerequisite, close_eyes, 8.717798, If you want to [sleep] then you should [close your eyes], none, none
dream, RelatedTo, sleep, 8.134126, [dream] is related to [sleep], none, none
dream, HasPrerequisite, sleep, 7.745967, Something you need to do before you [dream] is [sleep], none, none
rest, RelatedTo, sleep, 7.246241, [rest] is related to [sleep], none, none
sleep, HasPrerequisite, get_in_bed, 6.324555, Something you need to do before you [sleep] is [get in bed], none, none
going_to_bed, Causes, sleep, 6, Sometimes [going to bed] causes [sleep], none, none
gathering_energy_for_tomorrow, HasSubevent, sleep, 6, Something you might do while [gathering energy for tomorrow] is [sleep], none, none
resting, HasSubevent, sleep, 6, Something you might do while [resting] is [sleep], none, none
sleep, MotivatedByGoal, rest, 6, You would [sleep] because you want to [rest], none, none
sleep, RelatedTo, rest, 5.688233, [sleep] is related to [rest], none, none
sleep, HasPrerequisite, go_to_bed, 5.656854, If you want to [sleep] then you should [go to bed], none, none
dream, HasSubevent, sleep, 5.656854, One of the things you do when you [dream] is [sleep], none, none
having_rest, Causes, sleep, 5.291502, Sometimes [having a rest] causes [sleep], none, none
being_tired, CausesDesire, sleep, 5.291502, [Being tired] would make you want to [sleep], none, none
sleep, RelatedTo, bed, 5.039444, [sleep] is related to [bed], none, none
snore, HasPrerequisite, sleep, 4.89898, Something you need to do before you [snore] is [sleep], none, none
snoring, HasPrerequisite, sleep, 4.89898, [snoring] requires [sleep], none, none
sleep, RelatedTo, night, 4.720593, [sleep] is related to [night], none, none

Fig. 4. The output txt file for the term “sleep”.

Crawling for information about a particular new word

Input: new word X

Output: txt file contains the word and its relations

New records in the structure data base with the new word information

Step1: Run the URL of the English dictionary API, **http://api.conceptnet.io/c/en/X,**

Step2: Download the JSON file of the word

Step3: Using the class library “Newtonsoft.Json” Do

- extract the start node
- extract the end node
- extract the weight from the @id
- extract the relation sentence

Step4: Prepare txt file contains the word information and relations

Step 5: - Store the txt file

- Store the information in the structured database

Fig. 5. Crawling Algorithm.

in addition to information about the words (e.g. the word type, synonyms, prerequisite verbs, events of each script and its possible sequences). The main purpose of the relational database is to improve the retrieval time of a script's information using SQL queries. Fig. 7 shows the main tables in the database.

As seen in Fig. 7, the following tables exist in the internal database:

- **ScriptsNames**: contains the name of each script and its ID.
- **Scripts:** retains the detail of each script and stores the IDs of edges connecting events, such that each cognitive script is presented as a tree of events connected by edges. The details of these events are found in Events table.
- **Terms:** dictionary of all cognitive scripts.
- **Events:** contains the verbs, subjects, objects, and the predicate form of the events.
- **all_edges:** represents the information retrieved from the ConceptNet knowledge base.

For clarification, the Terms table is searched first using an SQL query then the output of the query is linked to the all_edges table to retrieve the corresponding information for the term in hand. If a term is not found, the system automatically crawls ConceptNet to find information about this term, download and save this information in the internal database for future searches.

4. MAP-and-REDUCE TECHNIQUE

Map-and-reduce is a programming model and an associated implementation for processing big datasets in parallel to reduce the processing time. The use of the map-and-reduce technique for web query retrieval has been discussed in many works

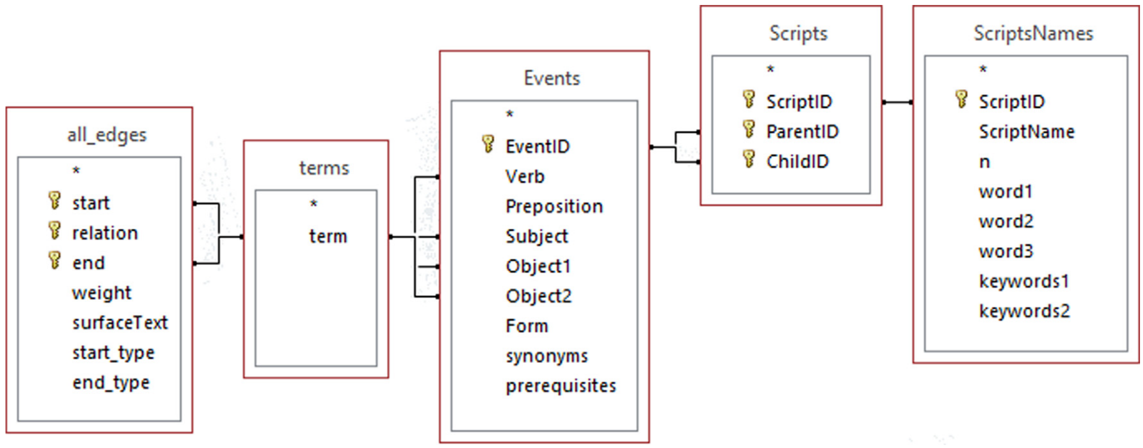

Fig. 6. Structured tables in the internal database.

Table: ScriptsNames

ScriptID	ScriptName
1	carbuying
2	classical
3	concert
4	cinema
5	stadium
6	pharmacy
7	restaurant
8	apprehending
9	order_a_drink

Table: Scripts

ScriptID	ParentID	ChildID
1	101	102
1	101	103
1	102	104
1	103	104
1	104	105
1	105	106
1	106	107
1	107	109
1	109	110

Table: Terms

term
arrive
look
see
talk
learn
recommend
decide
tell
accept

Table: Events

EventID	Verb	Preposition	Subject	Object1	Object2	Form
101	arrive		customer	car dealer		arrive(customer, car dealer)
102	look	at	customer	car		look_at(customer, car)
103	see		agent	customer		see(agent, customer)
104	talk	to	agent	customer		talk_to(agent, customer)
105	learn	about	agent	customer		learn_about(agent, customer)
106	recommend		agent	car		recommend(agent, car)
107	decide	to	customer	buy	car	decide_to(customer, buy, car)
109	tell		agent	customer	price	tell(agent, customer, price)
110	accept		customer	price		accept(customer, price)

Table: all_edges

start	relation	end	weight	surfaceText	start_type	end_type
arrive	RelatedTo	come	2.289978	[arrive] is related to [come]	none	none
look	RelatedTo	see	5.707889	[look] is related to [see]	none	none
look	IsA	appearance	2	[look] is a type of [appearance]	noun	noun
see	MannerOf	receive	2	[see] is a way to [receive]	verb	verb
talk	RelatedTo	speech	2.635906	[talk] is related to [speech]	none	none
talk	RelatedTo	speak	5.129912	[talk] is related to [speak]	none	none
learn	RelatedTo	knowledge	3.161012	[learn] is related to [knowledge]	none	none
learn	HasPrerequisite	study	4.472136	If you want to [learn] then you	none	none
recommend	Synonym	commend	2	[recommend] is a synonym of	verb	verb
decide	RelatedTo	choose	2.990652	[decide] is related to [choose]	none	none
tell	Synonym	assure	2	[tell] is a synonym of [assure]	verb	verb
tell	MannerOf	inform	2	[tell] is a way to [inform]	verb	verb
accept	RelatedTo	take	2.350319	[accept] is related to [take]	none	none

Figure 7 Screen shots of the internal tables

Fig. 7. Screen shots of the internal tables.

[31,34,35] as a mean to provide fast and accurate retrieval. This work provides a new application for map-and-reduce in knowledge retrieval that is the retrieval of cognitive scripts.

The retrieval process of cognitive scripts using map-and-reduce technique is shown in Fig. 8. The process unfolds as follows: the target script is analyzed, tokenized, and transformed into the predicate form (all scripts in the corpus are already tokenized and are represented in the predicate form). A bag-of-words for each script is created to be later used by the map-and-reduce technique in the retrieval process. The retrieval algorithm is presented in Fig. 9.

Map-Reduce for retrieving cognitive scripts

Input: **Target Script S**

Output: the most relevant script **T**, with the highest similarity to **S**

Step1:build the set of bag of words for script **S, such that, B(S)=**$\sum_{i=1}^{n} w_i$,
where n = # words in (S)

Step2: for each script T_j in the corpora **C** do

- **Build** bag of words for $T_j = B(T_j)$
- **Mapper:** map each word (w_i) such that $\forall (w_i \vee s_i) \in B(S) \wedge B(T_j)$ **mapped into** $<w_i, v_k>$, where v is the frequency value of word w_i in node **k** of Script T_j
- **Reducer:** reduce similar $<w_i,v_k>$ into one tuple $<w_i, \sum_k v_k>$
  $\forall w_i \in B(T_j)$ calculate the term frequency $TF(w_i)=\frac{\sum_k v_k}{|B(T_j)|}$
- **calculate the similarity** $Sim(S,T_j)= \sum_i TF(wi)$

Step 3: Store the similarity values of all scripts

Step4: **pick the** T_j with the highest $Sim(S,T_j)$

Step 5: output T

Fig. 9. Map-and-Reduce Retrieving Technique.

4.1. Illustrative example

Suppose that the user entered the “concert” script shown in Fig. 10, the map-and-reduce technique is applied as follows:

- Step 1: base script S=”concert”, the bag of words of S is

B(S)={buy, ticket, online, print, show, enter, stadium, snacks, seating area, sit_on, chair, stand, listen_to, song, sing, dance, stop, performing, go_to, backroom, go, home, counter, take, warm_up, say, welcome, start}

- Step 2: assume the first script in the corpus is the “cinema” script which is presented Fig. 11;

T1 = cinema script, then

- B(T1) = {buy, ticket, enter, theater, buy, popcorn, enter, auditorium, sit_down, watch, movie, be, scared, end, stand_up, go, home, turn_off, show, start}
- Mapper will map the common words of B(S) and B(T1) into < key, value > form as:

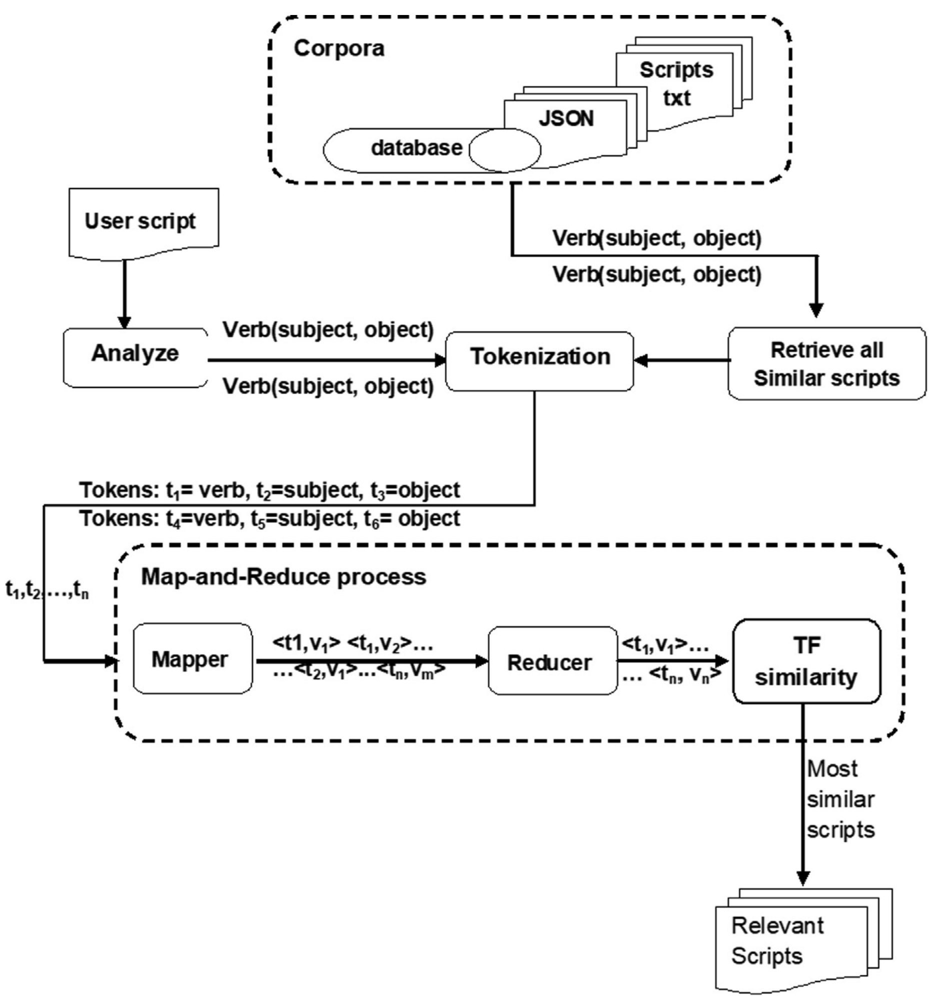

Fig. 8. The retrieval process.

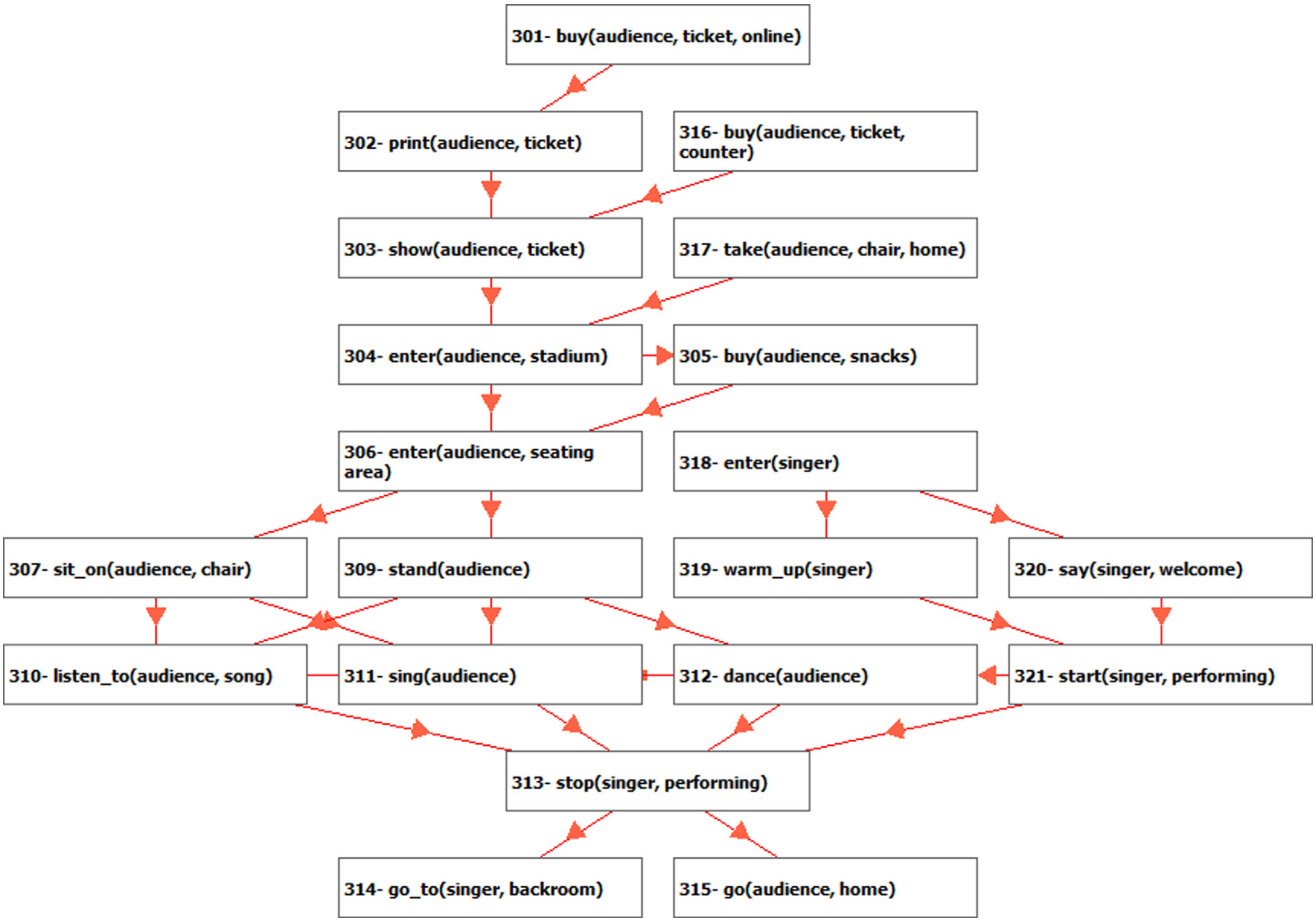

Fig. 10. Concert Script.

< buy,1 > node 401| <buy,1 > node 403| <ticket,1 > node 401| <show,1 > node 412| <enter,1 > node 402| <enter,1 > node 404| <go,1 > node 410| <home,1 > node 410| <start,1 > node 413.

- Reducer does the following:

Sort and reduce the number of pairs as follows:
<buy,2> <ticket,1> <show,1> <enter,2> <go,1> <home,1> <start,1 > calculate the TF for each word

word	buy	ticket	show	enter	go	home	start	Total
TF	0.1	0.05	0.05	0.1	0.05	0.05	0.05	0.45

- calculate the similarity

$Sim(S,T1)= \sum_1 TF(w1)$
$Sim\ (S, T_1) = 0.45$

5. Experimental results and analysis

For the purpose of this study, we adopted the same corpus used in [17] and added a null script for clarity of results. The database has 131 nodes and more than 262 words, Table 2 shows the scripts' names and the number of nodes in each script. All scripts were pre-processed and stored in a semi-structured data form as explained earlier in Section 2. The results obtained by the map-and-reduce in this work is compared to the results obtained by Pharaoh in [17]. The following subsections present the accuracy and time performance of the retrieval process for both techniques.

5.1. Accuracy measurements

This section presents the results from retrieving cognitive scripts by the map-and-reduce technique on the same multi-branched scripts using C#2020 programming language. Table 3a shows the results from the human participants [17]. It is worth mentioning that all the retrieved scripts are valid scripts, but some are more similar than others, that is why order was not considered in evaluating the results [17]. For the purpose of this study, a null script has been added to the corpus to indicate the absence of a relevant script.

Table 4 shows the results from Pharaoh [17] and our results from the application of the map-and-reduce technique. In order to compare the performance of both techniques, the retrieved scripts are compared to the human results in Table 3 and a confusion matrix is built.

As seen in Table 5, the map-and reduce technique provided higher accuracy, precision, recall, and specificity over Pharaoh. The harmonic mean (F1) shows the precedence of the map-and-reduce technique over Pharaoh. This shows that the map-and-reduce technique with ConceptNet can be effectively used for contextual retrieval of similar cognitive scripts to a query script.

5.2. Time performance measurements

For the purpose of this comparative study, we implemented Pharaoh [17] so that we can compare its cost to map-and-reduce using the same hardware, operating system, and programming language. This subsection focuses on comparing the retrieval time for Pharaoh and map-and-reduce with and without threads. As shown

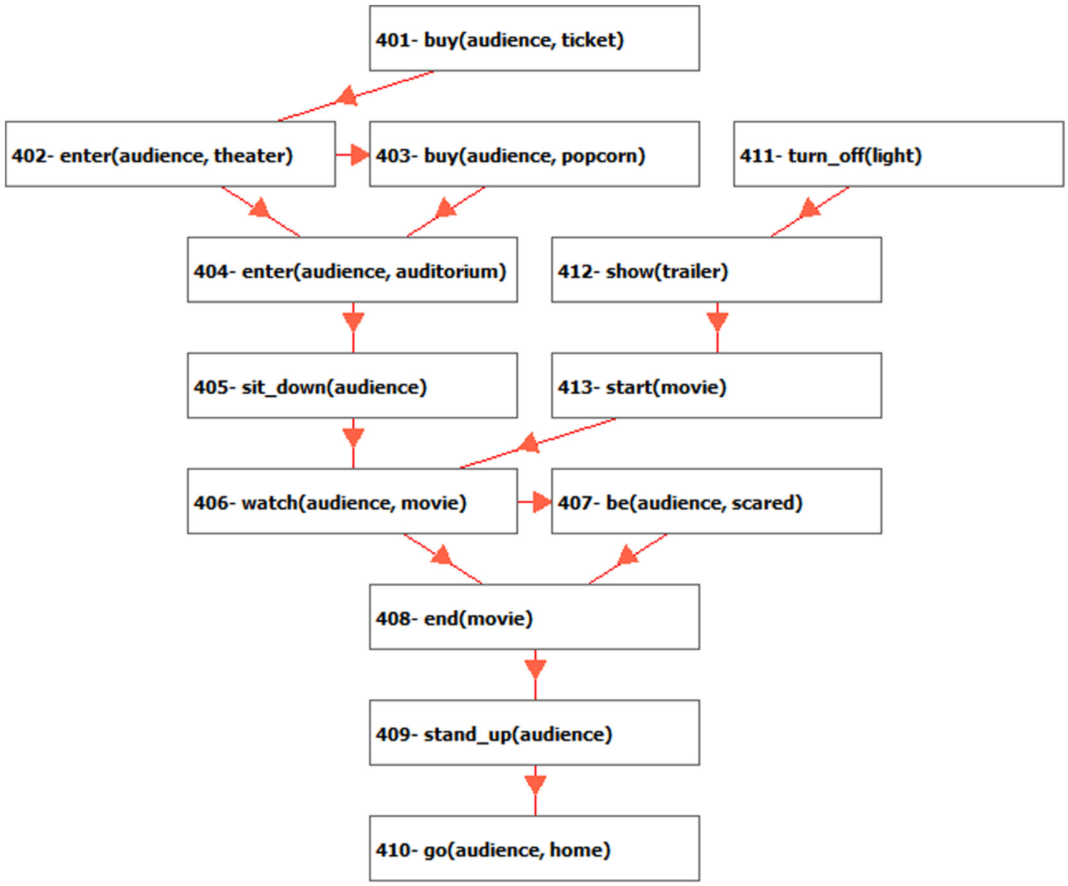

Fig. 11. Cinema Script.

Table 2
The stored scripts for evaluation.

IID	Name	#nodes
Scr1	Apprehending	7
Scr2	Car_buying	16
Scr3	Cinema	13
Scr4	Classical	17
Scr5	Concert	20
Scr6	Order_a_drink	10
Scr7	Pharmacy	12
Scr8	Restaurant	21
Scr9	Stadium	15
Scr10	Null Script	0

Table 3
Retrieved scripts based on human participants.

Query Script	Human Participants Selected scripts		
apprehending	null script	null script	null script
car_buying	pharmacy	order_a_drink	null script
Cinema	stadium	concert	classical
classical	concert	cinema	stadium
concert	classical	Cinema	stadium
order_a_drink	restaurant	pharmacy	car_buying
pharmacy	order_a_drink	restaurant	car-buying
restaurant	order_a_drink	pharmacy	car_buying
stadium	cinema	concert	classical

Table 4
Retrieved scripts based on Map-Reduce & Pharaoh.

Query Script	map-and-ReduceTechnique			Pharaoh		
apprehending	null	Null	Null	null	Null	Null
car_buying	Order_a_drink	Cinema	Stadium	Pharmacy	Null	Null
Cinema	Stadium	Classical	Concert	Stadium	Concert	Classical
Classical	Concert	Cinema	Stadium	Concert	Cinema	Stadium
Concert	Classical	Stadium	Cinema	Classical	Cinema	Stadium
Order_a_drink	Restaurant	Pharmacy	Stadium	Stadium	Concert	Cinema
Pharmacy	Restaurant	Order_a_drink	car_buying	Restaurant	car_buying	Null
Restaurant	Order_a_drink	Pharmacy	car_buying	Pharmacy	Null	Null
Stadium	Cinema	Concert	Classical	Cinema	Concert	Classical

in Table 6, Pharaoh suffers from high retrieval time which makes it an expensive algorithm to use, especially with large size scripts and large corpora. On the other hand, the map-and-reduce technique has shown low retrieval time which makes it a cost-effective algorithm to use with large corpora of scripts. For example, Table 6 shows that the retrieval time for similar scripts to the target ‘apprehending script’ using Pharaoh took more than three seconds, while took less than 1 s using map-and-reduce.

For further investigation, we implemented both techniques using threads to see how much threads would expedite the retrieval pro-

Table 5
Confusion matrix for the most 3 similar scripts retrieving process.

		Predict (Map-and-Reduce)		Predict (Pharaoh)	
		True	False	True	False
Actual (Expert opinion)	TTrue	TP(24)	FN(3)	TP(20)	FN(7)
	FFalse	FP(3)	TN(51)	FP(7)	TN(47)
		Map-and-Reduce	Pharaoh		
Accuracy	$\frac{TP+TN}{TP+TN+FP+FN}$	92.6%	82.7%		
Precision	$\frac{TP}{TP+FP}$	89%	74%		
Recall (Sensitivity)	$\frac{TP}{TP+FN}$	89%	74%		
F1	$2*\frac{Precision*recall}{precision+recall}$	89%	74%		
Specificity	$\frac{TN}{TN+FP}$	94%	87%		

Table 6
Retrieval time for cognitive scripts using non-thread programming.

Cognitive Scripts	Pharaoh(ms)	Map-and-Reduce(ms)
Apprehending	3101	630
Car_buying	3206	725
Cinema	3156	677
Classical	3153	757
Concert	3140	739
Order_a_drink	3075	660
Pharmacy	3239	700
Restaurant	3045	697
Stadium	3167	667
Average Time	3142	694

Table 7
Retrieval time for cognitive scripts using threads.

Cognitive Scripts	Pharaoh(ms)	Map/Reduce(ms)
Apprehending	1724	350
Car_buying	1790	311
Cinema	1673	336
Classical	1801	291
Concert	1905	374
Order_a_drink	1650	272
Pharmacy	1633	261
Restaurant	1660	298
Stadium	1715	338
Average Time	1728	315

cess. Table 7 shows how using threads cuts the time taken by the both techniques to almost half the actual time reported in Table 6.

The results from Table 6 and Table 7 are promising as they show the ability of the map-and-reduce technique to successfully retrieve similar cognitive scripts to a query script in a very short time, especially with using threads as it took just 10% of the original time Pharaoh took. This indicates how map-and-reduce can be used effectively with large scripts and/or large corpora. These results open new arena for SIAs to find similar BTs and learn from them to enrich their experiences and allow for more complex behaviors.

6. Conclusion

Behavioral trees (BTs) have been widely used to represent complex behaviors of social intelligent agents (SIAs). Many works have focused on the use of BTs with restricted set of actions to allow for socially accepted behavior by SIAs or the prediction of future behavior in a particular domain or scene. However, there was no much consideration for possible learning that can occur between similar domains, which might allow for the enrichment of these BTs and for new behaviors to emerge rather than relying on a pre-canned set of actions. Choosing the right representation for BTs and appropriate retrieval process of similar BTs are keys to allow for that learning process to occur.

Cognitive scripts can be used to represent BTs and allow for them to evolve through the allocation and retrieval of similar ones that SIAs can learn from. Retrieval of contextually similar cognitive scripts poses the challenge of not only finding the right script but also retrieving this script in a reasonable time, especially when a large-size scripts and/or large corpuses are considered. This paper examined a novel representation of BTs as cognitive scripts and the use of the map-and-reduce technique to retrieve contextually similar cognitive script in a very short time as a first step in the learning process that can occur between BTs. The results were compared to those obtained from applying Pharaoh, which is to the extent of our knowledge, the only algorithm used in the past to contextually retrieve cognitive scripts.

A two-proportion z-test was performed to determine if the proportion of accuracy was significantly greater for the map-and-reduce method over Pharaoh using R. The results indicate that the proportion of accuracy for map-and-reduce is significantly greater than the proportion of accuracy for Pharaoh with p-value = 0.05 and p-value = 0.13 for both techniques, respectively. Other results show that the map-and-reduce technique overrules existing algorithms like Pharaoh in terms of accuracy, specificity, and sensitivity, in addition to be computationally cheap as it reduces the computing time by 80% of the actual time taken by Pharaoh.

The results of this work show that the representation of behavior trees as cognitive scripts can aid in the contextual retrieval of similar BTs, which SIAs can use to learn from and enrich their current experiences. Future work will focus on the implementation of a learning process between BTs represented as cognitive scripts, the development of an evaluation model for newly generated BTs, and the application of behavior prediction techniques on the retrieved cognitive script(s) to aid in predicting future behaviors in similar domains.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgement

This research was funded by the Deanship of Scientific Research at Princess Nourah bint Abdulrahman University through the Fast-track Research Funding Program.

References

[1] Flórez-Puga, G., Gomez-Martin, M., Diaz-Agudo, B., & Gonzalez-Calero, P., "Dynamic expansion of behaviour trees. In Proceedings of Artificial

Intelligence and Interactive Digital Entertainment Conference", AAAI Press, pp. 36-41, October 2008.
[2] Francillette, Y., Bouchard, B., Bouchard, K., & Gaboury, S, "Modeling, learning, and simulating human activities of daily living with behavior trees", In Knowledge and Information Systems, pp.1-32, 2020.
[3] Li, B., Lee-Urban, S., Appling, D. S., & Riedl, M. O., "Crowdsourcing narrative intelligence", Advances in Cognitive systems, Vol. 2, No. 1, 2012.
[4] Tîrnăucă C, Montaña JL, Ontañón S, González AJ, Pardo LM. Behavioral modeling based on probabilistic finite automata: An empirical study. Sensors May 2012;16(7):958.
[5] Nicolau M, Perez-Liebana D, O'Neill M, Brabazon A. Evolutionary behavior tree approaches for navigating platform games. IEEE Trans Comput Intell AI Games 2017;9(3):227–38.
[6] Toubman, A., Poppinga, G., Roessingh, J. J., Hou, M., Luotsinen, L., Løvlid, R. A., ... & Turcanık, M. (2015, November). Modeling cgf behavior with machine learning techniques: Requirements and future directions. In Proceedings of the 2015 Interservice/Industry Training, Simulation, and Education Conference (pp. 2637-2647).
[7] Diller, D. E., Ferguson, W., Leung, A. M., Benyo, B., & Foley, D., "Behavior modeling in commercial games. In Proceedings of the 2004 Conference on Behavior Representation in Modeling and Simulation (BRIMS)", pp. 17-20, May 2004.
[8] Kamrani F, Luotsinen LJ, Løvlid RA. In: Learning objective agent behavior using a data-driven modeling approach. IEEE; October 2016. p. 002175–81.
[9] Luotsinen LJ, Kamrani F, Hammar P, Jändel M, Løvlid RA. In: Evolved creative intelligence for computer generated forces. IEEE; October 2016. p. 3063–70.
[10] Chaari T, Ejigu D, Laforest F, Scuturici VM. A comprehensive approach to model and use context for adapting applications in pervasive environments. J Syst Softw 2007;80(12):1973–92.
[11] Bouchard K, Ajroud A, Bouchard B, Bouzouane A. In: SIMACT: a 3D open source smart home simulator for activity recognition. Berlin, Heidelberg: Springer; 2010. p. 524–33.
[12] Bouchard, B., Gaboury, S., Bouchard, K., & Francillette, Y. , "Modeling human activities using behaviour trees in smart homes", In Proceedings of the 11th PErvasive Technologies Related to Assistive Environments Conference, pp. 67-74, june 2018.
[13] Dey, R., & Child, C., "Ql-bt: Enhancing behaviour tree design and implementation with q-learning", In 2013 IEEE Conference on Computational Intelligence in Games (CIG) (pp. 1-8). IEEE, August 2013.
[14] Zhang Q, Yao J, Yin Q, Zha Y. Learning behavior trees for autonomous agents with hybrid constraints evolution. Applied Sciences 2018;8(7):1077.
[15] Bazire M, Brézillon P. In: Understanding context before using it. Berlin, Heidelberg, July: Springer; 2005. p. 29–40.
[16] Hodhod, R., & Magerko, B.," Pharaoh: Conceptual blending of cognitive scripts for computationally creative agents", In the Twenty-Seventh International Flairs Conference., May 2014.
[17] Hodhod, R., Magerko, B., & Gawish, M., "Pharaoh: Context-based structural retrieval of cognitive scripts", International Journal of Information Retrieval Research (IJIRR), Vol. 2, No.3, pp. 58-71, 2012.
[18] Speer, R., & Havasi, C., "Representing General Relational Knowledge in ConceptNet 5", In LREC, pp. 3679-3686, May 2012.
[19] Tamine-Lechani, L., Boughanem, M., & Daoud, M.," Evaluation of contextual information retrieval effectiveness: overview of issues and research", Knowledge and Information Systems, Vol. 24, No.1, pp. 1-34, 2010.
[20] Yanchang Fu, Long Qin and Quanjun Yin. A Reinforcement Learning Behavior Tree Framework for Game AI, in the proceeding of the 2nd International Conference on Economics, Social Science, Arts, Education and Management Engineering (ESSAEME 2016), pp. 573—579, 2016.
[21] Sarker IH, Colman A, Han J, Khan AI, Abushark YB, Salah K. Behavdt: a behavioral decision tree learning to build user-centric context-aware predictive model. Mobile Networks and Applications 2020;25(3):1151–61.
[22] Sarker IH. A machine learning based robust prediction model for real-life mobile phone data. Internet of Things 2019;5:180–93.
[23] Tseng, E., & Braginsky, D. (2018). "Providing context relevant search for a user based on location and social information", U.S. Patent No. 9,978,022. Washington, DC: U.S. Patent and Trademark Office
[24] ELMougi AM, Omar YM, Hodhod R. An Evaluation Model for Auto-generated Cognitive Scripts. International Journal of Advanced Computer Science and Applications (IJACSA) 2019;10(8):333–40.
[25] Deligiannis, P., Loidl, H. W., & Kouidi, E., "Improving the diagnosis of mild hypertrophic cardiomyopathy with mapreduce", In Proceedings of third international workshop on MapReduce and its Applications Date, pp. 41-48, June 2012.
[26] Savage N. Better medicine through machine learning. Commun ACM 2012;55 (1):17–9.
[27] Mantha, P. K., Luckow, A., & Jha, S.," Pilot-MapReduce: an extensible and flexible MapReduce implementation for distributed data", In Proceedings of third international workshop on MapReduce and its Applications Date, pp. 17-24, June, 2012.
[28] Geery D. Reducing the storage burden via data deduplication. Computer 2008;41(12):15–7.
[29] Sethi P, Kumar P. Leveraging hadoop framework to develop duplication detector and analysis using Mapreduce, Hive and Pig. In: In 2014 Seventh International Conference on Contemporary Computing (IC3). p. 454–60.
[30] Ye K, Jiang X, He Y, Li X, Yan H, Huang P. Hadoop: A scalable Hadoop virtual cluster platform for MapReduce-based parallel machine learning with performance consideration. In: In 2012 IEEE International Conference on Cluster Computing Workshops. p. 152–60.
[31] Liao, J., Zhao, Y., & Long, S., "MRPrePost—A parallel algorithm adapted for mining big data", In 2014 IEEE Workshop on Electronics, Computer and Applications, pp. 564-568, IEEE, May 2014.
[32] Assunçao, J., Fernandes, P., Lopes, L., & Normey, S, "Distributed Stochastic Aware Random Forests–Efficient Data Mining for Big Data", In 2013 IEEE International Congress on Big Data, pp. 425-426, June 2013.
[33] Hanson, K., "Cognitive script theory. Encyclopedia of Children", Adolescents, and the Media, pp. 186-188, 2007.
[34] Chawda, B., Gupta, H., Negi, S., Faruquie, T. A., Subramaniam, L. V., & Mohania, M. K., "Processing Interval Joins on Map-Reduce", In EDBT, pp. 463-474, 2018.
[35] Hashem IAT, Anuar NB, Gani A, Yaqoob I, Xia F, Khan SU. MapReduce: Review and open challenges. Scientometrics 2016;109(1):389–422.

FIRST AUTHOR Safia Abbas is an associate professor in the Department of Computer Science, Faculty of Computer and Information Sciences, Princess Nourah Bint Abdulrahman University, KSA since 2019 and University of Ain Shams, Cairo, Egypt during 2016-2018. During 2006-2011, she received her Ph.D. from the Graduate School of Science and Technology, Niigata University, Japan. A strong theme of her work is in swarm optimizers, security in cloud, and medical diagnosis using machine learning and data mining.

SECOND AUTHOR Rania Hodhod is an associate professor and assistant chair at the TSYS School of Computer Science, Columbus State University (CSU) in USA. She has a Ph.D. in Computer Science from University of York, UK. She has an extensive background in artificial intelligence, machine learning, intelligent systems, and serious games as well as 70+ peer-reviewed publications in these areas. She is the holder of CSU International Educator of the Year in 2020, CSU Research Scholarship Award in 2019, and CSU Teaching and Learning Scholarship Award in 2017. She was a University System of Georgia Chancellor's Learning Scholar from 2018-2020.

THIRD AUTHOR Mohamed Abd el-Salam El-Sheikh is an Assistant Professor and Assistant Chair at Basic Science Department, Cairo University, Egypt. He has a Ph. D. in Computer Science from Cairo University, Egypt. He holds a Master's degree in Computer Science from Helwan University, Egypt and a Bachelor's degree in Physics and Computer Science.