Automated Reasoning in the Age of the Internet

Automated Reasoning in the Age of the Internet[⋆]

Alan Bundy[0000−0002−0578−6474], Kwabena Nuamah[0000−0002−6868−9858], and Christopher Lucas[]

School of Informatics, University of Edinburgh
A.Bundy@ed.ac.uk, k.nuamah@ed.ac.uk, c.lucas@ed.ac.uk

Abstract. The internet hosts a vast store of information that we cannot and should not ignore. It's not enough just to retrieve facts. To make full use of the internet we must also infer new information from old. This is an exciting new opportunity for automated reasoning, but it also presents new kinds of research challenge.

– There are a huge number of potential axioms from which to infer new theorems. Methods of choosing appropriate axioms are needed.
– Information is stored on the Internet in diverse forms, e.g., graph and relational databases, JSON (JavaScript Object Notation), CSV (Comma-Separated Values) files, and many others. Some contain errors and others are incomplete: lacking vital contextual details such as time and units of measurements.
– Information retrieved from the Internet must be automatically curated into a common format before we can apply inference to it. Such a representation must be flexible enough to represent a wide diversity of knowledge formats, as well as supporting the diverse kinds of inference we propose.
– We can employ forms of inference that are novel in automated reasoning, such as using regression to form new functions from sets of number pairs, and then extrapolation to predict new pairs.
– Information is of mixed quality and accuracy, so introduces uncertainty into the theorems inferred. Some inference operations, such as regression, also introduce uncertainty. Uncertainty estimates need to be inherited during inference and reported to users in an intelligible form.

We will report on the FRANK[1] system that explores this new research direction.

Keywords: Query answering · Prediction · Automated reasoning · World Wide Web

[⋆] This work has been funded by a University of Edinburgh studentship for the second author and Huawei grant HIRP O20170511.

[1] Formally know as RIF: Rich Inference Framework. We changed the name as the RIF acronym is already in use, standing for Requirements Interchange Format.

1 Introduction

We describe the FRANK (Functional Reasoning Acquires New Knowledge) system. FRANK applies inference to knowledge sources on the World Wide Web to derive estimates of new information and reliably assigns an uncertainty to it. It applies deductive, arithmetic and statistical reasoning to the results of information retrieval. We call this *rich inference*. An earlier description appeared in [9]. FRANK's main focus is on estimating the values of numeric attributes, but it sometimes returns qualitative answers, e.g., the query "Which country will have the largest population in Africa in 2021?" returns the name of the African country with the maximum estimated population.

Our hypothesis is:

> *A combination of information retrieval with deductive, arithmetic and statistical reasoning can be used accurately to estimate novel information and to assign a reliable uncertainty estimate to it.*

To address the issues raised in the abstract above, we have adopted the following techniques:

- The knowledge required to answer a query is retrieved from a wide variety of different knowledge sources on the Web. We employ APIs for each of the common knowledge formats in order to match the knowledge sought to the knowledge sources from which we retrieve it.
- This knowledge is then dynamically curated into a common format and stored in a query-specific ontology. This enables our inference operations to combine knowledge from diverse sources. Our common format is *alists*, i.e., sets of attribute/value pairs (see Definition 1). Alists can also be interpreted as n-ary, typed, logical relations (and sometimes also as functions), where $n + 1$ is the size of the set, the compulsory *Predicate* attribute's value is the predicate of the relation and the other attribute names are the types (see Definition 1). These pairs are both extracted from the particular knowledge item, e.g., the *Subject*, *Predicate* and *Object* attributes, and also augmented with attribute values from the source itself, e.g., the *Time*, *Units* and *Uncertainty* attributes. Alists provide the flexibility we need to cope with relations of diverse type signatures.
- Queries are represented as conjunctions of alists. Some of their attributes' values will be logical variables, whose value is unknown when the query is posed and which it is intended will be instantiated to a concrete value as a side effect of inference. Some of the variables in the query alist will be instantiated and returned as the answer to the query.
- FRANK's inference constructs a search tree with both AND and OR branches. Nodes are labelled with (sub-)goals represented as alists; the root node is labelled with the original query. Arcs are labelled with inference rules that enable a parent alist to be inferred from its child alists, i.e., inference is backwards from the root query to the leaf facts. If the search is successful, then the search tree will contain a proof as a subtree, which will contain

only AND branches. This proof tree will provide just those inference steps required to prove the query. During this proof, the query's variables will be instantiated to provide the required answer.

– The variables associated with the leaf alists of the proof tree are instantiated by matching them to facts stored in knowledge sources. The values of variables in parent alists are calculated by applying arithmetic aggregation operations to some of the variables in their child alists. The variables whose instantiated values are projected from child to parent are distinguished as *projection variables* (see Definition 2).
– Projected numeric values are assumed to have a Gaussian distribution and are returned as a mean and standard deviation. The mean is regarded as the answer and the standard deviation as an error bar on this answer. Aggregation operations are applied to both mean and standard deviation as they are inherited from leaf to root. Leaf nodes are assigned uncertainty values associated with the knowledge source from which they are taken. Knowledge sources are initially assigned default uncertainties, but these uncertainties are incrementally adjusted by a Bayesian process which compares the compatibility of rival sources of the same knowledge. Some inference operations also add additional uncertainty that is inherent in their nature, e.g., regression/extrapolation.

2 Alists: A Common Knowledge Format

Each node of the FRANK search tree is labelled by an association list or *alist*, which is a set of attribute/value pairs[2]. For example, the assertion that the population of the UK in 2011 is 63,182,000 people is represented by:

$$\{\langle Subject, UK\rangle, \langle Predicate, Population\rangle, \langle Object, 63,182,000\rangle, \langle Time, 2011\rangle\} \tag{1}$$

Alists enable FRANK to represent relations of any arity and with whatever types of arguments are required by the application. For example, alist (1) represents the ternary relation:

$$Population(UK, 63,182,000, 2011)$$

where the type signature of *Population* is:

$$Population : Subject \times Object \times Time \mapsto Bool$$

So, alists can be seen just as a syntax for typed logical formula and deduction with them as a logical inference process. All the different knowledge formats used in the knowledge sources accessed by FRANK can be curated into alists.

[2] See `https://en.wikipedia.org/wiki/Association_list` accessed on 5.6.18. Alists are not lists but sets, but the 'alist' terminology has, unfortunately, become standard.

2.1 Definition of Simple Alists

We can formalise a simple alist as follows:

Definition 1 (Simple Alist) *A simple* alist *is a set of pairs* $\{\langle A_i, a_i\rangle | 1 \leq i \leq n\}$*, where each* A_i *is an attribute and* a_i *is its value. This will sometimes be written as* $\{\langle A_1, a_1\rangle, \ldots, \langle A_n, a_n\rangle\}$ *or abbreviated as* $\mathcal{A}$*.*

- *We will use the notation* $\mathcal{A}(t)$ *to indicate that* $\mathcal{A}$ *contains a distinguished term* t *at some unspecified redex.*
- *We will use the notation* $\mathcal{A}[A] = a$*, when* $\langle A, a\rangle \in \mathcal{A}$*, i.e., that* a *is the value of attribute* A *in* $\mathcal{A}$*.*
- *We will use the notation* $\mathcal{A}[\boldsymbol{b}/\boldsymbol{a}]$ *to indicate that the values* $\boldsymbol{a}$ *of some attributes* $\boldsymbol{A}$ *are pairwise replaced by* $\boldsymbol{b}$ *in an alist* $\mathcal{A}$*.*
- *One attribute must be* $Predicate$*. This allows the alternative representation of:*
$$\{\langle Predicate, P, \rangle, \langle A_1, a_1\rangle, \ldots, \langle A_n, a_n\rangle\}$$
as $P(a_1, \ldots, a_n)$ *where* $P : A_1 \times \ldots \times A_n \mapsto Bool$*.*

Typical attributes are *Subject*, *Object*, *Predicate*, *Time*, etc. Values can be names, numbers, functions, etc. *Object* values are often numbers, but not exclusively so.

2.2 Variables in Alists

A (sub-)goal alist usually has some attribute values that are variables. During proof search, these variables may be instantiated. Variables in leaf alists are instantiated by being matched against facts stored in knowledge sources. Projection variables are instantiated to values that are passed from child alists to their parents. Each alist has an aggregation operation attribute with a function value h, say. This function h is applied to the projection variables of the child alists to instantiate the projection variable of the parent. This aggregation operation is associated with the inference rule on the AND branch connecting the parent to its children. The aggregation operation enables each alist to be regarded, not just as a relation, but also as a function from projection variables of the children to the projection variable of the parent.

The various variables appearing in an alist are defined as follows:

Definition 2 (Projection, Auxiliary and Operand Variables) *Let* $\mathcal{A}$ *be an alist.*

- *Its* projection variables *are the variables whose values are to be projected from it to its parents. They are prefixed with a* $?$*, e.g.,* $?x$ *denotes a projection variable. In general, an alist may have several projection variables, so we use vector notation to denote them all, e.g.,* $\boldsymbol{?x}$*.*

- *Its* auxiliary variables *are the variables whose values are used locally within $\mathcal{A}$, but are not projected to its parents. They are prefixed with a \$, e.g., $\$x$ denotes an auxiliary variable. In general, an alist may have several auxiliary variables, so we use vector notation to denote them all, e.g.,* $\boldsymbol{\$x}$.
- *Its* operand variables *are the variables that are used as arguments for $\mathcal{A}$'s* aggregation operation h. *An operand can be either a projection or an auxiliary variable but must exist as an attribute value in $\mathcal{A}$.*

By distinguishing projection variables, we can also view alists as *functions*, which return the value(s) of the projection variable(s) as their results. This view of alists is crucial in formalising the propagation of projection variables (see §5) and the treatment of nested queries (see §2.3).

A query or (sub-)goal is represented as an alist containing projection variables, e.g., if we want to ask what the population of the UK was in 2011, then the query would be:

$$\{\langle Subject, UK\rangle, \langle Predicate, Population\rangle, \langle Object, ?p\rangle, \langle Time, 2011\rangle\} \tag{2}$$

where $?p$ is a projection variable which will be projected up.

2.3 Nested Queries and Alists

Some queries are *nested*, e.g., "What was the GDP in 2010 of the country predicted to have the largest total population in Europe in 2018?". FRANK's initial formalisation of nested queries is to represent them as *compound alists*, i.e., alists which have alists as some of their values. If alists are viewed only as relations, then nesting one relation inside of another would be a syntax error. It does make syntactic sense, however, if the nested alists are given their *functional* interpretation. That is, the inner alist returns the values of its projected variables as the value of an attribute of the outer alist. We can represent the situation abstractly as:

$$\{\ldots, \langle Attribute_1, \{\ldots \langle Attribute_2, ?x\rangle, \ldots\}\rangle, \ldots\} \tag{3}$$

where the projection variable value $?x$ of the attribute $Attribute_2$ of the inner alist becomes the value of the attribute $Attribute_1$ of the outer alist.

For FRANK's inference system to apply, however, such compound alists need to be normalised into conjunctions of simple alists. In the case of compound alist (3), normalisation gives the following conjunction of two simple alists.

$$\{\ldots \langle Attribute_2, ?x\rangle, \ldots\} \wedge \{\ldots, \langle Attribute_1, \$x\rangle, \ldots\}$$

Note that x does not necessarily become a projection variable of the outer alist, so we have used $\$x$ here, rather than $?x$.

3 Curation and Enrichment

Curation is a bridge between the diverse knowledge source formats and the target common format used by FRANK. The leaf alists in the search tree are sub-goals that must be translated into the format used by the knowledge source being queried and then matched to the knowledge in that source. Matching instantiates variables in the sub-goal alist. FRANK incorporates APIs for each of the knowledge formats used by the knowledge sources that it queries. It also incorporates information retrieval procedures for each type of knowledge source, e.g., SQL, SPARQL, JSON, OWL.

FRANK uses a variety of KBs for (1) finding synonyms of terms in lookup decompositions, (2) finding sub-parts of geographical entities in geospatial decompositions and (3) retrieving facts about entities. KBs used include Wordnet [7], Geonames [11], Wikidata [12], ConceptNet [6], Google Knowledge Graph [10], and the World Bank's datasets on country development indicators[3].

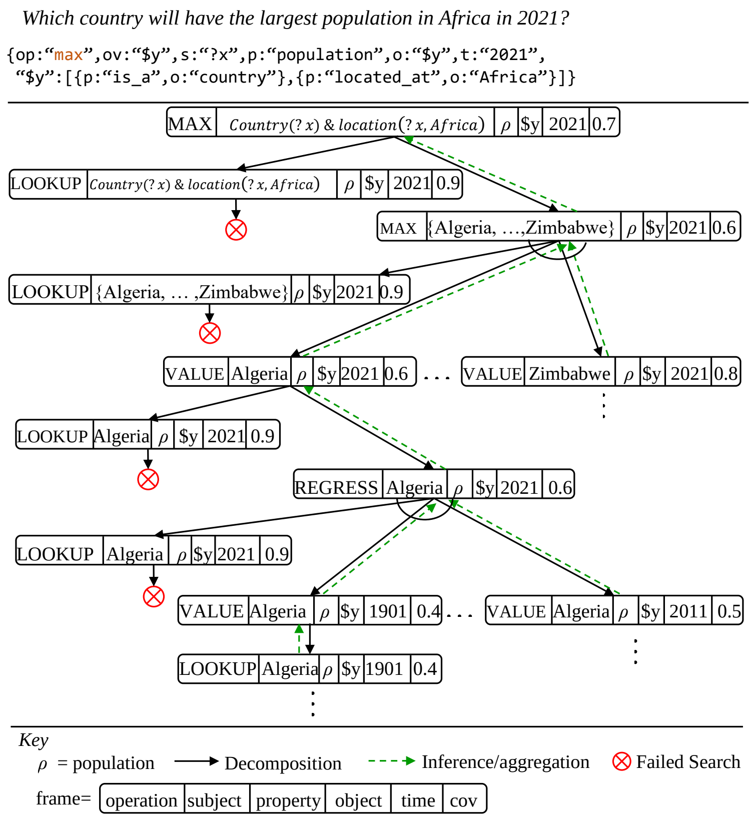
Some knowledge formats have restricted functionality, e.g., representing only unary or binary relations, e.g., only a predicate between a subject and an object. The leaf alist, however, may represent an n-ary relation for $n > 2$, and some of these additional attributes may contain variables that must be instantiated, e.g., units, time and uncertainty. The additional fields can often be found as global properties of the knowledge source, e.g., a car manufacturer may express all dimensions as centimetres, census data will record the year of the census, FRANK will have a record of the uncertainty it currently assigns to each knowledge source. These global properties enable variables in these additional attributes to be given values.

4 Search and Proof Trees

FRANK's inference can be represented as an AND/OR search tree. The OR branches represent the different ways in which FRANK may attempt to prove a sub-goal. Only one of these branches needs to succeed in order for the sub-goal to be proved. The AND branches represent the different child sub-goals that all need to be proved in order to prove the parent sub-goal. An example search tree is given in Fig. 1.

– Each node in the search tree is a box labelled by a truncated representation of its alist. The arcs between nodes represent inference operations. AND branching is represented by a circular line connecting the branches. OR branches have no such line.
– The first word in each alist is the aggregation operation. For instance, LOOKUP returns the value to which projection variable(s) have been instantiated by information retrieval; VALUE returns the value of its unique child's alists' projection variable(s); MAX returns the maximum value of the

[3] `https://data.worldbank.org/`



```
{op:“max”,ov:“$y”,s:“?x”,p:“population”,o:“$y”,t:“2021”,
 “$y”:[{p:“is_a”,o:“country”},{p:“located_at”,o:“Africa”}]}
```

Which country will have the largest population in Africa in 2021?
MAX | Country(?x) & location(?x, Africa) | ρ | $y | 2021 | 0.7
LOOKUP | Country(?x) & location(?x, Africa) | ρ | $y | 2021 | 0.9
MAX | {Algeria, …,Zimbabwe} | ρ | $y | 2021 | 0.6
LOOKUP | {Algeria, … ,Zimbabwe} | ρ | $y | 2021 | 0.9
VALUE | Algeria | ρ | $y | 2021 | 0.6
VALUE | Zimbabwe | ρ | $y | 2021 | 0.8
LOOKUP | Algeria | ρ | $y | 2021 | 0.9
REGRESS | Algeria | ρ | $y | 2021 | 0.6
LOOKUP | Algeria | ρ | $y | 2021 | 0.9
VALUE | Algeria | ρ | $y | 1901 | 0.4
VALUE | Algeria | ρ | $y | 2011 | 0.5
LOOKUP | Algeria | ρ | $y | 1901 | 0.4
Key
ρ = population
Decomposition
Inference/aggregation
Failed Search
frame= operation | subject | property | object | time | cov

Fig. 1. FRANK's Search Tree for the query "Which country will have the largest population in Africa in 2021?"

children's projection variables; REGRESS returns the result of extrapolating, to a new x value, a function formed by regression on the children's $\langle x, y \rangle$ pairs.

– FRANK always tries direct look-up first. Only if this fails does it apply an inference operation. $\otimes$ represents a failure, e.g., look-up failed because no matching fact could be found in any knowledge source.

– The main inference operations used are geospatial and temporal decomposition. Geospatial decomposition breaks a $Subject$ into parts, applies the query to each part and then combines the results, e.g., by summing them or taking the maximum. Temporal decomposition applies the query to different (often older) time values, applies regression to form a function and then applies that function to the original time.

– A successful search tree contains a proof sub-tree. This is indicated by the dotted arc lines in Fig. 1.

– In Fig. 1, after failure to find the query's answer by direct look-up, geospatial decomposition is applied to apply the query directly to each African country and then to return the country whose population is the maximum. Direct look-up of each country's population in 2021 fails, so temporal decomposition is applied to census data for each country from the years 1901 to 2011. Regression is applied to this data to form a graph, which is then extrapolated to 2021. Since the AND branching rates are quite high, ellipsis has been used to compact the search tree to readable dimensions.

5 Inference and Aggregation

FRANK's current inference operations are information retrieval and the geospatial and temporal decomposition rules, which have been described in §4 above. Plans to extend these are outlined in §10.1. A unique property of FRANK's inference is its combination of deductive reasoning with statistical reasoning. In particular, it forms functions by regression, which provides the ability to reason about functions: second-order deduction, such as calculus.

An aggregation operation is associated with each application of an inference operation. Aggregation propagates the values of instantiated projection variables from child alists to parent alists, and so back to the root alist, where it becomes the mean value of the answer to the original query.

Geospatial Decomposition: Depending on the query, the values of the children's projection variables can be aggregated by various arithmetic operations, such as finding: the maximum or minimum; the mean, median or mode; the sum or product; or the number of children. If there are only two children, then we can also find whether the first is equal to, greater than or less than the second.

Temporal Decomposition: Each of the children's alists returns a $\langle x, y \rangle$ pair. Regression is applied to these values to form a function f. This f is extrapolated or interpolated to a new value of x by applying this function to it and returning the corresponding $f(x)$ as the parent's projection variable value.

6 Uncertainty

It is important that some measure of uncertainty is associated with the results returned by FRANK. Knowledge obtained from the Web is of variable quality, depending on the reliability of the source. Moreover, some of the inference operations we use, e.g., regression, contribute additional uncertainty. FRANK must keep track of this uncertainty and report it to user, so that they know how much to trust the result. We propose error bars as the best measure of uncertainty to assign to the kind of numerical results estimated by FRANK.

– Probabilities do not work. For instance, the probability of $\exists ?p.\ Population(UK, ?p, 2025)$ is 1, i.e., it is certain that the UK will have some population count in 2025. What we need to know is the accuracy of the value FRANK assigns to $?p$. Due to the inherently vagueness of population counts, the probability that any one value of $?p$ is absolutely correct is essentially 0 — or, more accurately, the question is inherently meaningless unless we know what value to assign to people who are in the process of dying, being born or in a vegetative state, etc., and what instant in time the census was taken.

– What we really need is to give a *range* for the answer. Error bars are a well known way of expressing such ranges, that many people will have seen on graphs, etc. They are also standard in numerical science.

Gaussian distributions (also known as bell curves) are ubiquitous in many numerical estimates. They can be defined by two measures: the *mean*, which gives an average of the distribution and the *standard deviation*, which describes the spread of the distribution, so is ideal to express the error bars. We have, therefore, adopted Gaussians as our distribution of uncertainty.

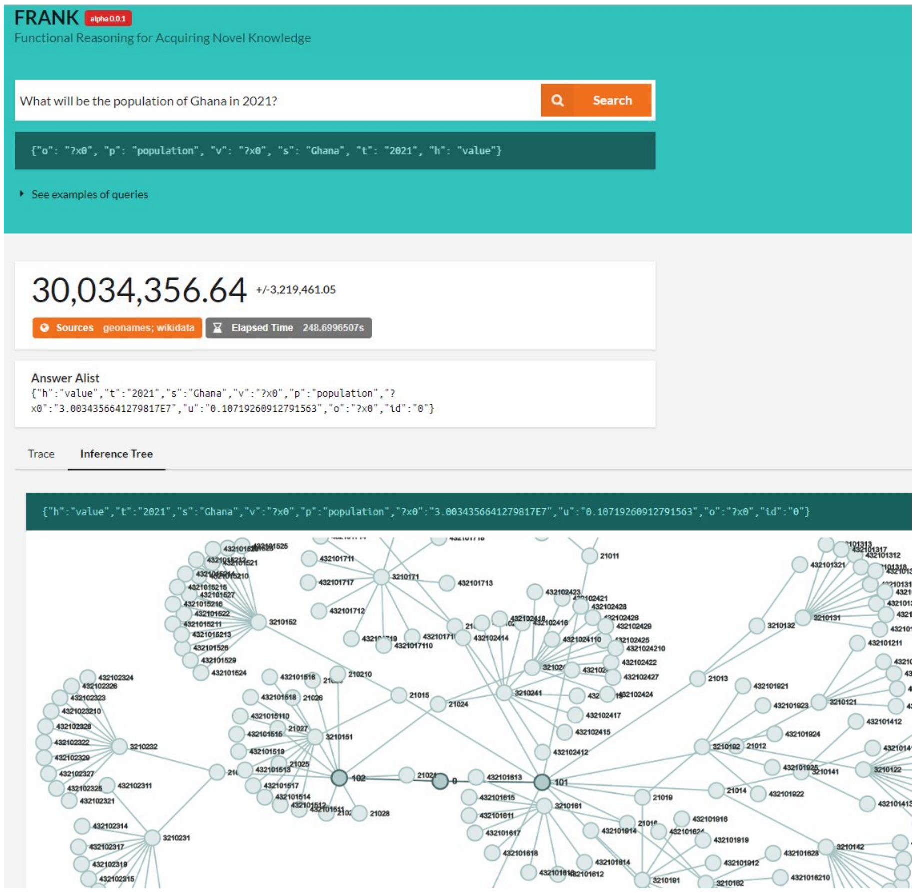
We return the mean value as our estimate of the value of a numerical projection function. The width of the error bar then gives a measure of the uncertainty associated with the mean. We use two different ways to express error bars. Firstly, we can use the standard deviation, which gives an absolute measure of the range of values of the projection variable that fall within the standard deviation. Secondly, we use the *coefficient of variation* (CoV). This is the mean divided by the standard deviation. It gives a relative measure of the range. For instance, we could turn the CoV into a percentage by multiplying it by 100, and then say that the mean was, say, within 5% of the correct[4] value. The CoV is ideal for propagating the uncertainty from leaf to root nodes. That's because the projection variables vary from node to node of the proof tree. So, the standard deviations are not comparable, but the CoVs are, so can be combined [2]. To report the final uncertainty back to the user, though, the standard deviation is sometimes preferable. It can be readily calculated by multiplying the CoV by the mean. For more details about the use of uncertainty in FRANK, see [8].

Note that this measure of uncertainty only applies to real numbered values. We are looking into measures of qualitative uncertainty as future work (see §10.2).

7 Interface

FRANK has a simple natural language interface. This enables users to type queries and receive answers in a restricted grammar of English via a GUI. The natural language processing employs the spaCy: off-the-shelf NLP library [5]. The grammar restrictions are to ensure that the query can be represented as an alist. A snapshot of FRANK's GUI is given in Fig. 2.

[4] Assuming that the correct value lies within one standard deviation. Since the potential range is infinite, this is a compromise between being informative and reasonabl accurate. One could, instead, use two or more standard deviations.



What will be the population of Ghana in 2021?

```
{"o": "?x0", "p": "population", "v": "?x0", "s": "Ghana", "t": "2021", "h": "value"}
```

30,034,356.64 +/-3,219,461.05

Sources geonames; wikidata
Elapsed Time 248.6996507s

Answer Alist

```
{"h":"value","t":"2021","s":"Ghana","v":"?x0","p":"population","?
x0":"3.0034356641279817E7","u":"0.10719260912791563","o":"?x0","id":"0"}
```

```
{"h":"value","t":"2021","s":"Ghana","v":"?x0","p":"population","?x0":"3.0034356641279817E7","u":"0.10719260912791563","o":"?x0","id":"0"}
```

Fig. 2. FRANK's GUI for the query "What will be the population of Ghana in 2021?"

- The question is typed in the query box at the top.
- This query is then translated into alist form, which is displayed in abbreviated form in the dark box immediately below the query.
- FRANK's answer of 30,034,356.64 is then displayed below this with a standard deviation of ±32119461.051857124 as the error bar.
- The instantiated root alist is given below this. Note that the uncertainty value here is the CoV, not the standard deviation, which only appears in the final answer.
- At the bottom is FRANK's search tree, in which the nodes labels are given as numbers to save clutter, but these can be unpacked by clicking on them. The search tree has a zoom option, so that the user can get an overview or examine one part in more detail.

This interface is currently in an early stage of development. This will include giving appropriate feedback to users who ask queries outwith FRANK's grammar. Eventually, we plan to deliver this interface as an open web service.

8 Evaluation

We have evaluated our hypothesis that:

> *A combination of information retrieval with deductive, arithmetic and statistical reasoning can be used accurately to estimate novel information and to assign a reliable uncertainty estimate to it.*

Our evaluation has two parts. Firstly, we want to know how accurately FRANK has estimated the answer. For instances, is the estimated answer within one standard deviation of the true answer? Secondly, we want to know how accurate our uncertainty estimates are. For instance, are the true errors proportional to the estimated errors. For both parts of this evaluation, we need to know the true answers. We do this by a 'leave one out' methodology. That is, our queries are of known values, but we prevented FRANK from looking the values up directly, forcing it to estimate them from other known values. We compared FRANK's success rate with two comparator query answering systems: Google search and Wolfram|Alpha. These comparators were not prevented from direct look-up[5].

We randomly generated a set of 100 queries using property terms related to the country indicators in the World Bank data-set. We used 60 of these queries as a training set during the development of FRANK and used the remaining 40 for the test set. These 40 were grouped into four types:

Retrieval: Queries whose answers were found by direct look-up. FRANK was not prevented from direct look-up for these queries.

Inference Queries: Simple queries where several facts needed to be combined by inference but where regression was not needed.

Nested Queries: Compound queries that had to be normalised, but where regression was not needed.

Prediction: Queries for which regression and extrapolation/interpolation were required.

Table 1 shows a favourable comparison of FRANK's percentage success rate to two popular query answering systems: Google Search and Wolfram|Alpha[6], that also use the World Bank's dataset. A result is counted as a success if it is within one standard deviation of the true answer. FRANK performs better than both its two comparators on all four query types but, as might be expected, it did especially well when predication was required, since no prediction answers were pre-stored.

[5] Mainly because we couldn't do so, so they did have an advantage over FRANK.

[6] `https://www.wolframalpha.com`

Queries	Google Search(%)	Wolfram|Alpha(%)	FRANK(%)
Retrieval	70	80	90
Aggregation Queries	20	70	80
Nested Queries	-	50	80
Prediction	10	20	70
Average %	25	55	80

Table 1. Evaluation results by query types, showing the percentage of queries answered successfully.

Fig. 3 is a scatter plot to compare actual error to estimated error. On the y axis is the ratios between (a) the absolute difference between the true and estimated values and (b) the true value. On the x axis is the estimated error represented by the CoV. Ideally, this scatter plot would approximate a straight line, showing that actual and estimated error were proportional. The dotted line is the best fit straight line to these points. This is a fair fit to the data.

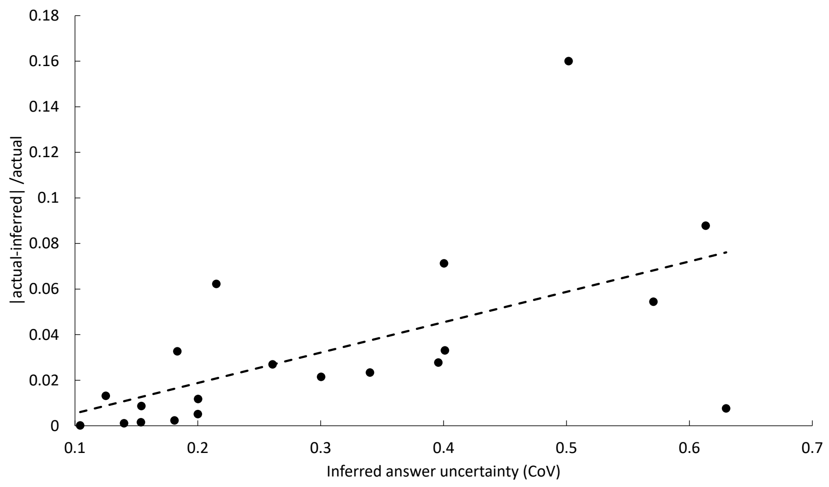


Fig. 3. Comparison of estimated error against actual error

9 Related Work

We have found nothing quite like FRANK to compare it to. The best fit is probably the first author's previous work on the GORT system [1]. This system solved guesstimation problems, where an approximate answer was required to a numeric problem, e.g., "how many cars, parked bumper to bumper, would be needed to reach from Edinburgh to Glasgow?". It also searched the Web for facts and inferred new information from it, but its inference operations were limited to simple arithmetic and its error bars just showed the range of different answers these methods had found.

Table 1 showed a favourable comparison of FRANK's performance to two other popular query answering systems. We plan further such comparisons, but inference-based query answering systems, e.g., [4], have gone out of fashion since they cannot operate at web-scale. So we did not find a lot of modern systems against which to compare FRANK. IBM's Watson [3] was too finely tuned to solving questions in the game show, Jeopardy!. As a result, it could not be directly compared to FRANK.

Currently, the field of information retrieval[7] is focused on extracting known information from the Web. Its main research challenge is interpreting queries in natural language (mostly, but not exclusively, English). FRANK's simple NL interface is described in §7, but this is not the focus of our research.

Given our focus on the inference of new information from old, Table 2 gives a comparison of FRANK to traditional work in the automation of reasoning.

FRANK	**Automated Reasoning**
Lots of uncertain information	A few, certain axioms.
Lots of facts, few rules	More rules than facts
Diverse formats	Uniform format
Diverse inference operations	Deductive inference
Depleted information	All information present
Killer app: query answering	Killer app: formal verification

Table 2. Comparison of FRANK and Automated Reasoning

10 Future Work

FRANK is still under active development and we have plans to extend it in several directions.

10.1 Generalising Decomposition Rules

FRANK currently uses only two decomposition rules: temporal and geospatial, but there is the potential for many more. The general form of a decomposition rules is given in Definition 3.

[7] `https://en.wikipedia.org/wiki/Information_retrieval` (accessed 4.7.18)

Definition 3 (Decomposition Rule) *A decomposition rule is an implication of the form:*

$$\begin{aligned} &Decompose(\mathcal{A}(\boldsymbol{x}), \tau) = [\mathcal{A}_j | 1 \leq j \leq m] \\ &\qquad \implies \mathcal{A}[\boldsymbol{h_\tau}(\epsilon \boldsymbol{?x_1}.\ \mathcal{A}_1(\boldsymbol{?x_1}), \ldots, \epsilon \boldsymbol{?x_m}.\ \mathcal{A}_m(\boldsymbol{?x_m}))/\boldsymbol{x}] \end{aligned}$$

where:

– $[\mathcal{A}_j | 1 \leq j \leq m]$ *is a form of list composition, that we have invented, which is analogous to set comprehension (as used in Definition 1 for instance).*
– $\boldsymbol{h}$ *is the inference operation that takes the values* $\epsilon \boldsymbol{?x_j}.\ \mathcal{A}_j(\boldsymbol{?x_j})$ *assigned to the projection variables* $\boldsymbol{?x_j}$ *of the child alists* $\mathcal{A}_j$ *and calculates the value* $\boldsymbol{h}(\epsilon \boldsymbol{?x_1}\ \mathcal{A}_1(\boldsymbol{?x_1}), \ldots, \epsilon \boldsymbol{?x_m}.\ \mathcal{A}_m(\boldsymbol{?x_m}))$ *of the variable* $\boldsymbol{x}$ *of the parent alist* $\mathcal{A}$*.*
– *Decompose is a function that takes the parent alist* $\mathcal{A}$ *and the type of decomposition* τ *and returns a list of* m *child alists* $\mathcal{A}_j$*. A list, rather than a set, is required here, as the order of the arguments to* $\boldsymbol{h}$ *must be specified. Vectors would also work.*
– *Note that the implication is from left to right: the values of the projection variables of the child alists determine the values of the operands of the parent alist. But FRANK works backwards to build the proof tree from the goal alist to the leaf node alists, whose projection variable values are then looked up on the Web.*

Different decomposition rules can be generated by varying the definition of *Decompose*. For instance, Geospatial decomposition uses the $partOf$ hierarchies in various KBs to partition the value s of the $Subject$ attribute in $\mathcal{A}$. Currently, FRANK only uses this for breaking geographical regions into parts. It could equally well be applied to break a product into its components, e.g., to identify the most costly component.

Similarly, *isa* hierarchies could be used to identify the sub-types of an $Object$. This would be useful, say, to find the cheapest laptop meeting some minimal conditions on speed, memory capacity, etc.

We are currently exploring the space of potential decomposition rules and the applications they make possible.

10.2 Qualitative Uncertainty

CoVs provide a good method of assigning uncertainty to real-valued query answers and intermediate values used in their calculation. We plan to extend FRANK to non-numeric queries. Currently, FRANK is limited to non-numeric queries that involve only numeric calculations during aggregation, but with a final non-numeric answer, e.g., returning those members of a set that attain either a maximum or a minimum value on a particular numeric attribute. For these, we can use the CoV associated with the calculation that this value is indeed the extreme one. For instance, if the question is: "Which country will have the largest population in Africa in 2021?", then, although the answer will be a particular African country, we can assign to that answer the CoV associated with the calculation that its population is a maximum among the set of all African countries.

In this case, all the aggregation operations involved in the proof tree were arithmetic ones. We want to investigate how uncertainty values might be aggregated for the values of non-numeric projection variables. We will probably need a new uncertainty measure, as CoVs are associated with Gaussian distributions, which are fundamentally numeric. We will need to combine these new uncertainty measure with CoVs. We then need to associate appropriate aggregation operations to apply to non-numeric projection variables.

11 Conclusion

We have described the FRANK query answering system, which draws inferences from information on the Web to discover new information, including make predictions. FRANK is focused on numerical questions.

- The Web contains a huge and rapidly growing source of information. Despite the inherent uncertainty in this information, it is a source we can't afford to ignore.
- Merely retrieving known facts from the Web is to neglect most of its potential. We must infer new information from old. This is a job for automated reasoning.
- But this job raises a new range of challenges for the automated reasoning field.
  - It is necessary to locate the axioms needed from this huge store. FRANK's top-down proof search identifies the kind of axioms it needed, so that information retrieval can be used to find them.
  - The information we need is stored in a diverse number of formats. In order for automated reasoning to combine information in diverse formats, these must all be curated into a common format. FRANK uses alists, as they assimilate all the other formats.
  - Some source formats are overly restrictive, e.g., only allowing unary or binary predicates. Additional attribute values are often needed, e.g., time and units. Curation must also include finding these additional attribute values, so that they can match values in goal alists.
  - The inherent uncertainty in both knowledge sources and some inference methods must be inherited back up through the proof tree to provide the user with an uncertainty estimation for the answer that FRANK returns. FRANK propagates coefficients of variation: the standard deviation of the answer normalised by the mean. CoVs provide an error bar on the answer, which is returned as the mean. The propagated CoV is converted back to a standard deviation for the final answer, as this provides a numeric range in which the true answer is likely to fall.

– A user friendly interface is required for users to pose queries and receive answers. FRANK allows uses to pose questions in a restricted grammar of English.
– With these new challenges come exciting new opportunities.
  – Information retrieval is freed from simple factoid look-up, and can infer new information — even making predictions.
  – Inference can combine deduction, arithmetic and statistics. FRANK's evaluation shows that this combination of inference methods can both accurately estimate novel information and assign a reliable uncertainty estimate to it.

References

1. Bundy, A., G., S., Chan, M.: Solving guesstimation problems using the semantic web: Four lessons from an application. Semantic Web Journal (2013)
2. Clifford, A.A.: Multivariate error analysis: a handbook of error propagation and calculation in many-parameter systems. Applied Science Publ. (1973)
3. Ferrucci, D., Brown, E., Chu-Carroll, J., Fan, J., Gondek, D., Kalyanpur, A.A., Lally, A., Murdock, J.W., Nyberg, E., Prager, J., et al.: Building watson: An overview of the deepqa project. AI magazine **31**(3), 59–79 (2010)
4. Green, C.C., Raphael, B.: The use of theorem-proving techniques in question-answering systems. In: Proceedings of the 1968 23rd ACM national conference. pp. 169–181. ACM (1968)
5. Honnibal, M., Johnson, M.: An improved non-monotonic transition system for dependency parsing. In: Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing. pp. 1373–1378. Association for Computational Linguistics, Lisbon, Portugal (September 2015), `https://aclweb.org/anthology/D/D15/D15-1162`
6. Liu, H., Singh, P.: ConceptNet - a practical commonsense reasoning tool-kit. BT technology journal **22**(4), 211–226 (2004)
7. Miller, G.A.: Wordnet: a lexical database for english. Communications of the ACM **38**(11), 39–41 (1995)
8. Nuamah, K., Bundy, A.: Calculating error bars on inferences from web data. In: Intelligent Systems Conference. IEEE (2018)
9. Nuamah, K., Bundy, A., Lucas, C.: Functional inferences over heterogeneous data. In: 10th International Conference on Web Reasoning and Rule Systems (RR 2016). pp. 1–7 (June 2016)
10. Singhal, A.: Introducing the knowledge graph: things, not strings. Official Google Blog, May (2012)
11. Vatant, B., Wick, M.: Geonames ontology (2012)
12. Vrandečić, D., Krötzsch, M.: Wikidata: a free collaborative knowledgebase. Communications of the ACM **57**(10), 78–85 (2014)