Background ch:sota

Genetic Algorithms sec:ga GA is a non-deterministic population-based global search metaheuristic that reformulates optimization problems in a vocabulary that makes heavy usage of metaphors on biological evolution, thus also being called a nature-inspired metaheuristic.

GA typically represent solutions to a problem as bit vectors which correspond to the genotype in biological evolution. These genotypes encode a set of parameters which are passed to a so called fitness function which promotes a survival of the fittest and is thus the target of minimization or maximization in the underlying optimization problem.

A fixed size population of genotypes, in this context also called individuals is initially generated at random using a PRNG. This population of genotypes, parameterized by the population size (μ), is then iteratively evolved where the iterations are typically referred to as generations. In each generation, a subset of the population is selected to join a so called recombination pool, which is parameterized per the recombination pool size (λ). The selection of individuals is carried out using a so called selection operator. An important property of a selection operator is the selective pressure, which is informally described as the emphasis of selection on the best individual, where a high (small) selective pressure means a strong (weak) emphasis. We will discuss two of the most prominent selection operators further down in this text.

The individuals selected for recombination are then passed in pairs to a crossover operator, which generates a pair of childs by swapping the bit vectors representing the individuals at a given crossover pointNote that the literature contains a vast number of variations on the crossover operator such as multi-point crossoverspears:1990a and uniform crossover syswerda:1989a, however they are of no particular interest to this work. There has also been a debate over the usefulness of crossover operators which resulted in several extensive theoretical analyses with minimalistic toy problems with the result that they ,,can provably be useful”doerr:tcs2012a,spears:1991a.. The childs are then inserted into the next generations population together with the survivors, which are selected from the current population also using a selection operator.

A mutation operator is then applied to each individual in the next generation with a certain mutation probability (σ). The classic mutation operator mutates each selected individual by changing the bit value of a given bit in its bit vector representation with a probability denoted by the mutation rate (ρ), i.e. a mutation rate of $\rho = 12$ means that on average, half of the bits are changed to their opposing state.

These steps of selection, crossover and mutation are repeated until a certain termination criterion holds, which could include a maximum number of generations, a target fitness value or a test of population convergence.

Note that in the sense as defined above, GA were reportedlymitchell:1998a first conceived by John Holland in 1960 as an extension of general (μ+λ) Evolutionary Algorithms, the novelty being the introduction of crossover and selection operators whereas previous iterations of Evolutionary Algorithms only made use of mutation operators.

Selection Operators While the literature contains a great number of selection operators, for sake of brevity we will only introduce the two selection operators used in the remainder of this text, namely the Elitist Selectionmitchell:1998a and the Tournament Selectionmiller:cs1995a.

The Elitist Selection is a simple selection procedure which includes the N best performing individuals in the next generation. It is typically used in conjunction with another selection operator, as it would just lead to a stochastic hill climbing algorithm when used as only selection operator and there are much more efficient hill climbing algorithms, such as vaughan:ijc2005a for discrete or altman:cm1970a for continuous optimization problems.

The Tournament Selection is one of the most established selection operators, as it has been proven able to adjust the selective pressure independently from the population size and fitness function scaling-goldberg:1990a. It is parameterized by the tournament size k and the selection probability p. Initially, k individuals from the given population are randomly selected to enter a tournament. Then, the ith best individual in the tournament w.r.t. the fitness function is selected to win the tournament with the probability $p(1-p)^{i-1}$. This means that setting the probablity to $p = 1$ will result in a deterministic behaviour while setting $k = 1$ results in random selection.

Genetic Programming ssec:gp

GP is a subfield of GA which originated from the application of a GA in order to evolve computer programs in 1988 by John Kozakoza:1990a. While traditional GA are commonly used for numerical optimization and search problems, GP can be used for symbolic regression and classification.

In GP, programs are usually encoded as trees of operations and terminals, but other encodings have also been proposed. For example, Graff et al.graff:2016a use DAG to encode python programs, and Kvasnika and Pospchalkvasnieka:1998a introduced a condensed encoding of DAG, dubbed ,,Column Tables", which we will adapt and use in this work (see ssec:enrichmentgraph).

Multi Expresssion Programming ssec:mep

MEP, originally introduced by Ferreiraferreira:cs2001a as ,,Gene Expression Programming" denotes a special kind of GP where the genotype of an individual represents multiple solutions or subsolutions that has gained traction in recent years. Only the best amongst these multiple solutions is then used as the effective solution, also called the phenotype of the individual. This technique is used in problems where it does incur no additional cost to keep track of the fitness of subsolutions, such as is generally the case with the evaluation of a program represented as a tree: it can be easily seen that when evaluating such a tree-shaped program, one has to visit each node anyway in order to compute its result.

It can be used to evolve programs with the same complexity as traditional GP but much more efficient and excels in multiple-output problems. To this end it has been successfully applied to the TSPoltean:c2015a, data predictionzhang:i2013a, software effort estimationakram:c2018a, on-the-fly hyperparameter optimization for Evolutionary Algorithmsoltean:2003a and digital curcuit designoltean:2004b.

Semantic Genetic Operators ssec:sgo

Semantic Genetic Operators are a special kinds of crossover and mutation operators which take into account the semantics of a genotype rather than just discerning it as a bit vector without any further interpretation. They are therefore aware of the specific genotype encoding and manipulate the individuals w.r.t. the solution space rather than the encoding space.

Their recent successful application to a number of problemschen:2017a,forstenlechner:2017a,peng:ia2019a,suarez:rcs20
indicates that taking semantics into account is a promising novel direction for GP.

table[b] The Five Stars of Linked Open Data tab:fivestar tabularxp.15X¿p.25 Stars Requirements Example

Linked Data sec:ld

Linked Databerners-lee:2006a, bizer:2008a, bizer:ijsis2009a is a term coined by Tim Berners-Leeberners-lee:2006a in 2006. It refers to data available on the Web that is (1) structured, (2) uses URI as identifiers and (3) is interlinked with other data on the Web. Linked Data is enabled by the use of standard Web technologies such as the HTTP, the RDF and the SPARQL. It is closely related to the term Semantic Webberners-lee:sa2001a, shadbolt:iis2006a, which describes a Web of Data which, in opposition to the human-readable Web of Documents is specifically engineered to be machine-readable. The Semantic Web promotes a high interconnectedness of datasets by interlinking them with eath other, thereby enabling the ability to consume data from multiple datasets at once using semantic queries, e.g. using the SPARQL protocol.

When considering Linked Data published under open licenses, we speak of LOD. A quality scheme for LOD, namely the five stars of LOD, was given by Tim Berners-Lee in 2010berners-lee:2006a and is summarized in tab:fivestar.

Further Discussion of Linked Data?

The Resource Description Framework ssec:rdf

The RDF is a ,,standard model for data interchange on the Web"group:2014a. RDF models data in an abstract syntax which is defined in terms of an abstract graph-based data model. The nodes in this graph-based model belong to one of three atomic base types: IRI resources, literals and blank nodes. IRI resources and blank nodes denote a particular subject in the world, while literals only contain values which are subject to further interpretation. As blank nodes lack the notion of a global identifier, they serve as existential variables in the data model.

An RDF graph is a collection of statements, also called triples, which basically correspond to simple sentences in natural language. Each statement therefore consists of a subject, a predicate and an object. Every subject is either a IRI resource or a blank node, all predicates are IRI resources and objects can be either of the three basic types.

In the following, we will give a formal specification of an RDF dataset on which we will rely in the remainder of this work.

IRI RDF Dataset: Let R be the set of all RDF IRI resources, B be the set of all blank nodes and L be the set of all literals.

We call a set $D\{(s, p, o) \in (R \cup B) \times R \times (R \cup B \cup L)\}$ of triples an RDF dataset.

Note that for sake of simplicity, we will not distinguish between an RDF dataset and its abstract graph and therefore we from now on resort to referring to both as RDF datasets.

Vocabularies, Ontologies and Shapes ssec:onto While RDF specifies a basic set of vocabulary, it is not very expressive on its own and offers no mechanism to define vocabularies or data schemes.

In order to make statements about the nature of the data that is expressed, other standards which extend on the RDF semantics have to be used.

This is a concious decision as it allows to use multiple semantics, DL of different expressiveness to be used in conjunction with the RDF data model.

The two main standards that are used to that end are RDFS and OWL.

Contrary to the typical notion of ,,schema", as for example in XSD, RDFS is not used to define a closed schema in order to force adherence to it. Rather, it provides a set of standard predicates and resources that enable composing a simple ontology with mostly statements about class and predicate hierarchies, instances of classes and relationships between classes and predicates. It thereby enables the use of efficient reasoners for basic inference as well as more standardized tools for applications to discover and reason about what the data is about and how it is potentially shaped. The ,,potentially" in the last sentence is owed to the fact that RDF as well as RDFS semantics follow the open-world assumption, the absence of a fact is not enough to prove its negation holds true.

The OWL is a standard which allows for much more expressiveness compared to RDFS. There are three subsets of OWL: OWL Lite $\subset$ OWL DL $\subset$ OWL Lite. Their expressiveness increases from OWL Lite to OWL Full, which naturally comes at the cost of availability of efficient or even effective reasoning algorithms.

Another standard that can be used to accompany the RDF data model is the SHACL. Contrary to the previously introduced standards, SHACL semantics follow the Closed World Assumption which on one hand decreases its effective usability alongside RDFS and OWL but on the other enables an important application that can not be realized with the latter two, namely that of schema validation.

Linked Data Integration sec:ldi

Generally speaking, Linked Data Integration denotes a process of processing data whose goal it is to return one or many RDF dataset(s) for a specific use case, publication to the LOD Cloud or preparation for further usage by specific applications.

The input to Linked Data Integration can, but not necessarily must be Linked Data itself. Typical processing steps in Linked Data Integration include, but are not limited to (1) lifting, (2) linking, (3) fusion, (4) enrichment. Any one step could be left out, depending on the use case.

In the following, we will define each of these processing steps and give a narrow overview of the State of the Art.

Lifting ssec:lifting In the Linked Data community, lifting denotes the act of materializing RDF datasets from sources of non-linked data. These sources may be relational databases, proprietary file formats, measurements from sensor networks and any other structured or unstructured data that is not Linked Data.

The term is closely related to RDF mapping, whereas in mapping the focus is often on ad-hoc transformation of an evolving non-linked data source in contrast to the materializing approach in lifting

After lifting, the data is expected to be at least four-star data according to the scheme presented in tab:fivestar.

In stuhmer:2009a, the authors attempt to lift event data from the interaction with Web pages. Bizer et al.bizer:2010a introduced a framework for discovering mappings to relational data sources on the Web. In neumaier:2017a, ayed:2017a, schandl:2010a and fiorelli:2015a, the authors proposed approaches for lifting metadata on data portals, entries in the Docker registry, entries in file systems and whole spreadsheets, respectively. The approach in hasan:2011a considers a framework for continuously lifting sensor data to Linked Data in order to exploit the semantic interconnectedness of the sensor measurings for situation awareness.

Linking ssec:linking

The linking of RDF datasets, sometimes also called instance matching, amounts to finding a mapping M between two RDF datasets, commonly called the source dataset S and the target dataset T, such that the pairs $(s, t) \in M$ identified by the mapping abide to a given relation r.

This field is closely related to the instance matching and deduplication in databases. The main challenge here is that due to the very large number of instances in the Web of Data, the quadratic run time complexity of a brute force search can not be accepted.

To this end, it is necessary to derive algorithms that discard large numbers of potential pairs, preferably without losing the completeness or correctness property, all pairs (s, t) that abide by $r(s, t)$ should be in M and no pair in M should not abide by r.

A plethora of such approaches has been developed for textual datadrng14,xiao:p2008a,xiao:tds2011a,feng:vj2012a,isele geospatial dataradon$_2$017, $du : 2013a, shivaprabhu : 2017aaswellastemporaldataallenalgebra$.

Another challenge lies in the automatic configuration of so called link discovery tools.

A number of different approaches have been proposed.

The approaches in NGLY12,NGO+13b,isele:2013a use GA in an unsupervised, an active learning setting and both, respectively. In NGLY13, the authors use an altered grid search for unsupervised learning and finally sherif:2017a, the approach outperforming all of the previously mentioned, uses refinement operators for unsupervised as well as supervised learning.

Fusion ssec:fusion

The fusion of RDF datasets is a task in which two interlinked datasets that contain information about the same set of instances are merged to produce a joint dataset. To this end, an ontology mapping has to be generated in a first step. Subsequently, the fusion is carried out following a set of so called fusion rules that are either predefined by the system, the user or dynamically learned.

In nikolov:2007a,nikolov:2008a,nikolov:2008b, the authors use a fusion architecture called KnoFuss which handles the end-to-end fusion of RDF datasets. The apporach in mendes:2012a combines quality assessment and fusion of RDF datasets. A probabilistic framework is applied to the fusion problem indong:2014a and bryl:2014a is an approach to automatic learning of fusion rules in the setting of cross-language fusion from the Wikipedia corpus.

Enrichment ssec:enrichment

Enrichment is the least well-defined step in the Linked Data Integration lifecycle. Informally, enrichment corresponds to applying a set of modifications to a given RDF dataset such that it abides by certain use case-specific predefined criteria. We will call this set of modifications, addition and deletion of triples from the source dataset, an enrichment workflow in the remainder of this text. Examples of such modifications include the dereferencing of information from other datasets, the application of quality assurance methods to the dataset, the conformation of the used vocabulary or the filtering of its contents.

In practice this means, that RDF dataset enrichment is a problem that greatly differs from use case to use case. Notable applications of enrichment in the literature include abel:2011a, which considers the automatic extraction of interests from Twitter posts in order to enrich user profiles on the Social Web and choudhury:2009a, which applies a ranking method to the Youtube tag space in order to enrich datasets on the LOD Cloud with links to Youtube videos.

However, to our best knowledge, the only effort directed towards a general framework of RDF dataset enrichment together with an automatic learning approach can be found in sherif:2015a, where the authors propose a framework and learning algorithm called DEER. We will thus have a deeper analysis of DEER in the following section, where we will try to identify potential shortcomings of the framework in order to underline the necessity for our contribution in this work, DEER2.

DEER ssec:deer As we chose to build our approach based on the existing Open Source software DEER, we are going to highlight specific areas in the original paper resp. $software, whereweseechancesofsubstantialimprovement.$

*On Restrictiveness of Sequential Chaining The DEER models RDF dataset enrichment workflows as ordered sequences of enrichment functions. These enrichment functions take as input exactly one RDF dataset and return an altered version of it by virtue of addition and deletion of triples. It is argued that formally, any modification on a given dataset can be represented as a set of additions and deletions.

While this is certainly true, the application of two independent enrichment functions, that formally have no knowledge of each other, could be problematic as one function might delete triples which the other relies on in order to successfully apply the desired enrichment.

We therefore suggest replacing the simple sequential pipelining approach with a more sophisticated one based on arranging the enrichment functions in a DAG. In order to better distinguish our approach from the former, we will call atomic enrichment functions which allow for multiple inputs and multiple outputs enrichment operators.

*On Modularity and Attractiveness of the System The framework consists of five atomic enrichment functions that can be combined to form so called enrichment pipelines of arbitrary size. The language used implies that these enrichment functions are only examples and potentially more could be implemented.

In order to improve the real-world suitability of our contribution DEER2, we therefore will aim to attract developer interest by providing plugin facilities with low barriers to entry.

*On Appropriateness of Fitness Function The refinement operator-based learning approach in DEER is defined as the task of finding an appropriate enrichment pipeline given an input RDF dataset K and a target RDF dataset K'.

As the enrichment functions typically require manual configuration, the problem is twofold: it is not only the correct sequence of enrichment functions that is subject to the learning process but also the correct configuration of the enrichment functions that are part of the pipeline.

This is done by a combination of iterative construction of the enrichment pipeline and self-configuration of enrichment functions at each iteration.

That is, in order to decide which enrichment function should be next added to the current pipeline, each available enrichment function is self configured using the supplied training data. It is then applied to the result of the previous enrichment function and its result evaluated using a fitness function over the provided training data.

Note that one of the results of the evaluation of DEER was that it is sufficient to provide only a single training example. While it seems unusual for a learning method to rely on a single training example, it is a valid assertment in this specific case, as enrichment can be assumed to deal with more or less regularly shaped RDF datasets.

The dataset produced by the intermediate pipeline in each iteration is denoted as K_i with i being the position of the enrichment function producing K_i in the pipline.

The fitness function used is the F_1-scoresasaki:ttm2007a over the set of triples in K_i and K'.

We argue that this choice of fitness function is not optimal in a modular setting where solutions can be expected to be constructed in an iterative manner. We consider the following short example to illustrate the problem.

Let us assume without loss of generality that there exists one triple $t = (s, p, o) \in K$ for which we can intuitively identify at least one corresponding triple $t = (s', p', o') \in K'$. Then a first enrichment function might only alter our input triples predicates, yielding the intermediate result $t_1 = (s, p', o) \in K_1$. Now another enrichment function might alter the subject, yielding $t_1 = (s', p', o) \in K_2$ before we apply the final enrichment function and get to the desired result $t_3 = t' = (s', p', o') \in K_3$. While intuitively there are clearly improvements in each intermediate result w.r.t. our target triple t', a fitness function only measuring the correspondence between whole triples will not be able to detect these improvements.

We will therefore use an evenly weighted linear combination of the F_1-scores between all the subjects, all the predicates, all the objects and additionally all the whole triples in our input and target datasets.

*On Information Masking

A more subtle problem with the refinement operator-based approach in DEER lies in the combination of deterministic learning and self configuration. By assuming that the training example contains the necessary information for learning the pipeline, which is a sane assumption to make for regularly shaped input datasets, it is also implicitly assumed that it contains enough information for the self-configuration of all the enrichment functions.

As a trivial counterexample, consider an enrichment pipeline where we filter out all triples that contain a certain predicate p_f using the enrichment function e_3. This means that no triple (s, p_f, o) will be present in the target training data.

Let us assume that some triple (s', p', o') present in the target training data were constructed by another enrichment function e_2 from a triple $t_{masked} = (s, p_f, o)$ that was previously constructed by yet another enrichment function e_1.

Without the triple t_{masked} in the training data, e_1 might not be able to determine its own configuration and therefore will never get picked. In the end, our pipeline will never get learned using the deterministic bottom-up approach in DEER. We call the underlying problem information masking.

As a result of this analysis, we will use a non-deterministic approach based on GP in DEER2. We will complement this with an idea dubbed pre- and postcondition broadcasting which is developed in order to increase the chance of successful self configuration in the light of information masking.

Additional Preliminary Definitions

In this section we will introduce some additional definitions required to understand the remainder of this work which did not fit into one of the previous sections.

Simplified Concise Bounded Descriptions

Let D be an RDF dataset and $r \in D$ be an IRI resource in that dataset.

We define the neighbourhood $neighbours(r)$ of r in D as the set of triples that contain r as their subject. Formally, $neighbours(r)\{(s, p, o) \in D \mid s \equiv r\}$.

Let $k \in N > 0$. The SCBD $SCBD(r, k)$ of a resource r in D is then recursively defined per eq:cbd where $SCBD(r, 0)\emptyset$ and $SCBD(r, 1)neighbours(r)$.

equation SCBD(r, k) SCBD(r, k-1) $\cup \left(\bigcup_{(s,p,o) \in SCBD(r,k-1)} neighbours(o) \right) eq : cbd$