Linked Data as Stigmergic Medium for Decentralized Coordination

Torsten Spieldenner[1,2] and Melvin Chelli[1]
[1]*German Research Center for Artificial Intelligence (DFKI), Saarland Informatics Campus D3 2, 66123 Saarbrücken, Germany*
[2]*Saarbrücken Graduate School of Computer Science, Campus E1 3, 66123 Saarbrücken, Germany*

Keywords: Stigmergy, Stigmergic Medium, Coordination, Multi-Agent Systems, Linked Data, Semantic Web, Nature-inspired Algorithm.

Abstract: Algorithms inspired by nature have gained focus in research as a solution to classic coordination and optimization problems. A certain type of these algorithms employs principles of stigmergy: in stigmergic systems, coordination arises from agents leaving traces of their actions in the environment, or medium, that they work on. Other agents instinctively adapt their behavior based on the traces, by which, in the end, the fulfillment of a higher goal emerges from elementary actions of many, rather than thorough planning of complex actions of a few. Despite the perceivable uptake of stigmergic algorithms for coordination in various domains, a common clear understanding of a suitable digital stigmergic medium is lacking. It should however be assumed that a well-defined, properly modelled, and technically sound digital medium provides a crucial basis for correct, efficient and transferable stigmergic algorithms. In this paper, we motivate read-write Linked Data as generic medium for decentralized stigmergic coordination algorithms. We show how Linked Data fulfills a set of core requirements that we derived for stigmergic media from relevant literature, provide an application example from the domain of digital manufacturing, and finally provide a working example algorithm for stigmergic decentralized coordination.

1 INTRODUCTION

Coordination of tasks or resources is an ever prevalent topic of research in various domains, for example in traffic and public transport (Kanamori et al., 2014; Alfeo et al., 2018), or industrial production (Schraudner and Charpenay, 2020; Cicirello and Smith, 2004).

Among research that focuses on improving application of classic AI agent-based planning and optimization techniques, there is a specific trend that employs nature inspired algorithms to solve coordination and optimization problems (Chiong, 2009; Tzanetos et al., 2020). A re-occurring element from this class of algorithms is the use of concepts of *stigmergy* (Heylighen, 2015), a behavior observed, for example, in insect swarms: In a stigmergic system, participants of a process leave traces in their environment that influence subsequent behavior of other members of the swarm, such that a community behavior emerges that leads the swarm closer towards a common goal. Such nature inspired algorithms have been found a promising approach for more flexible, fault-tolerant, and scalable coordination of complex systems in above mentioned domains (Krieger et al., 2000; Ricci et al., 2007; Jevtić et al., 2012; De Nicola et al., 2020).

However, the majority of stigmergic systems in literature are highly use-case specific, or even implementation specific. It is in particular noticeable, that while focusing on specific algorithms and their implementations, existing work marginalizes or ignores the importance of a proper *medium*. The medium is the environment in which agents leave traces by performing actions, i.e., where stigmergic effects emerge, and it is considered the *"the mediating function that underlies the true power of stigmergy"* (Heylighen, 2006). The question of what a proper medium constitutes is particularly interesting in the world of AI optimization and coordination, where the medium has no tangible physical manifestation, but where agents operate entirely in an environment that is completely digital, while having only abstract correspondences to real world entities.

We believe that both research and application of stigmergic coordination algorithms can benefit greatly from a concise common understanding of a digital stigmergic medium. This medium should be provided by employing widely accepted open stan-

Spieldenner, T. and Chelli, M.
Linked Data as Stigmergic Medium for Decentralized Coordination.
DOI: 10.5220/0010518003470357
In *Proceedings of the 16th International Conference on Software Technologies (ICSOFT 2021)*, pages 347-357
ISBN: 978-989-758-523-4

dards to be independent of specific use-cases, domains, or technologies that implement the algorithm. It should be based on a well-defined, thoroughly formalized and established foundation to allow for soundly defined, general, transferable solutions.

Such a set of standardized and well-defined tools comes from the world of the *Semantic Web* (Berners-Lee et al., 2001). Based on the notion of *Linked Data* (Bizer et al., 2008) and typically modeled in terms of the *Resource Description Framework* (*RDF*) (Lassila et al., 1998) [1], the Semantic Web is commonly promoted as a generic integration layer for applications from various domains.

With the Internet, being the maybe largest digital medium that relies on a wide variation of stigmergic effects (Dipple, 2011), and being built around technologies and principles closely related to those of the Semantic Web, one may expect that a machine readable and writable Linked Data layer (*read-write Linked Data)* in the Semantic Web may as well provide a widely standardized, established, domain-independent interactive medium for stigmergy-based coordination (Dipple et al., 2013; Privat, 2012). To this end, this paper makes the following contributions:

We derive from relevant literature a *set of requirements towards digital media to serve as proper stigmergic medium*; we *establish read-write Linked Data as suitable medium* for stigmergic systems; and we *demonstrate the application* of read-write Linked data as stigmergic medium *with an example from the domain of virtual manufacturing*.

The remainder of the paper is structured as follows: We review relevant related literature in Section 2. In Section 3, we recapture core concepts of Linked Data architectures, as well as stigmergic systems. In Section 4, we derive requirements towards digital stigmergic media and discuss Linked Data as suitable choice. We demonstrate by example how to model a digital environment for stigmergic coordination in Section 5, and present a prototypical stigmergy-based coordination algorithm in 6. We discuss how by our chosen medium, the presented algorithm indeed achieves benefits of stigmergic systems in Section 7, and conclude our work and provide an outlook upon future challenges in section 8.

2 RELATED WORK

Stigmergic systems have been thoroughly described and analyzed (Heylighen, 2015; Dipple et al., 2013), and also been discussed with respect to applicability in Web-based environments (Dipple et al., 2014; Privat, 2012).

Agents in digital stigmergic systems communicate indirectly by reading from and writing to a shared data space, and by this can be considered to follow a *generative communication paradigm*. (Gelernter, 1985; Ciancarini et al., 1997). Practical implementations of this paradigm were, among others, realized in the System *Linda*, based on Tuple Spaces (Gelernter, 1985).

Stigmergy has been applied in various domains as a means for realizing coordination, for example in the field of robotics (Matarić et al., 2003; Krieger et al., 2000; Jevtić et al., 2012), or generally, distributed coordination in cyber-physical manufacturing (Cicirello and Smith, 2004; Schraudner and Charpenay, 2020). In the field of telecommunications (Bonabeau et al., 1998), concepts of evaporation rate and refresh rate from the ant world were applied for managing the dynamic requirements of a network. In the field of traffic and public transport, data acquired from large numbers of vehicles or passengers is used to predict hotspots and usage preference, and from this, plan more optimal routes for individuals based on the current situation (Kanamori et al., 2014; Alfeo et al., 2018).

Ant inspired optimization techniques have been in focus of research since the early 1990s, and various variations have been developed ever since (Dorigo and Blum, 2005; Heylighen, 2015). It has been shown that bio-inspired algorithms, such as stigmergy, perform generally well on decentralized decision making (Jevtić et al., 2012) as well as peer-recruitment and coordination for task accomplishments (Krieger et al., 2000). (Korošec et al., 2012) have shown a stigmergic approach for high dimensional numerical optimization (Differential Ant-stigmergy Algorithm, DASA). (Chopra et al., 2017) presented a distributed variant of the *Hungarian Method* for solving the Linear Sum Assignment Problem (LSAP), with multiple agents cooperating and finding an optimal solution for the LSAP without a central coordinator or shared memory. In recent years, the field of Multi-Agent Systems (MAS) has generally seen a growing interest in the concept of stigmergy (Yu and Cheng, 2020; Ricci et al., 2007; De Nicola et al., 2020; Cicirello and Smith, 2004).

In all the bio/insect-inspired works that have been discussed in this section, there has been no particular emphasis given to the medium to the extent that it is described by (Heylighen, 2015). Its fundamental meaning as a *shared environment* that agents use to leave traces, sense its state and act on it, has been largely overlooked. Some of the presented works (e.g. from the domain of traffic and transport) bypass the

[1] RDF 1.1 Primer document (Jan. 2021): https://www.w3.org/TR/rdf11-primer/

medium, and rely on direct agent-to-agent communication.

The fact that often the environment (or medium) is reduced to very basic representations, while agents lack cognitive abilities to make better use of a potentially more complex environment, has already been observed (Ricci et al., 2007). There is moreover an awareness of both the need for a common understanding of such an environment (Charpenay et al., 2020), and the difficulty to properly correspond concepts from the real-world scenarios to concepts in nature in order to properly employ those algorithms (Valckenaers et al., 2007). In this paper is, we discuss and present with appropriate examples, that read-write Linked Data layer is a suitable general stigmergic medium for said purposes.

3 BACKGROUND

3.1 Resource-Oriented Architectures

A Resource Oriented Architecture (ROA) is built around the notion of a *Resource* as common representation for and kind of virtual or real-world entities. (Fielding and Taylor, 2000; Lucchi et al., 2008). A resource is typically characterized by a name (identifier), its representation and links between resource representations (Lucchi et al., 2008). As defined by Fielding, a representation is a sequence of bytes and metadata to describe those bytes. A resource may be described by more than one representation at any given time i.e., provide the same content for example in different serializations formats. More detailed considerations of this architecture can be found in (Fielding and Taylor, 2000).

3.2 Linked Data Systems

Linked Data[2] is a way to share and structure information using links. It is a design principle that implements a Resource Oriented Architecture, and is built on the HyperText Transfer Protocol (HTTP) and HATEOAS (*Hypermedia as the Engine of Application State*) principles. That means that Linked Data enables software user-agents or applications to derive all necessary information to understand data, or how to interact with it, by following links as provided by the server (Fielding and Taylor, 2000). This understanding is supported by *semantic annotations* that describe to a data consumer the meaning of a particular field of data, or APIs that provide this data, thus leading to a uniform understanding of the data by different applications (Verborgh et al., 2011; Mayer et al., 2016).

These semantic descriptions are typically modeled using the *Resource Description Framework (RDF)* (Lassila et al., 1998), (see also footnote [1]).

RDF allows to formulate *statements* about *resources* in terms of triples that follow a *subject-predicate-object* structure. The *subject* symbolizes the resource, the *predicate* describes the qualitative aspect of the resource and/or describes the relationship between a *subject* and an *object*. A set of RDF triples constitutes a labeled *graph*, where the subject and object form nodes, connected by a directed edge (from subject to object) that is labeled via the predicate.

3.3 Stigmergic Media

Relevant results of research in the field of stigmergy-based self-organization have been very concisely summarized by Heylighen (Heylighen, 2015). We discuss stigmergy based on the findings from this paper exclusively. However, we would like to stress out that by this work being a very thorough survey over the topic that covers research from several decades, we well include findings from many different researchers with different view on the general topic.

Heylighen derives from his findings his own definition of stigmergy as an:

*"indirect, mediated mechanism of **coordination** between **actions**, in which the **trace** of an action left on a **medium** stimulates the performance of a subsequent action" (Heylighen, 2015, p. 5).*

The definitions of the core components of stigmergy as described in (Heylighen, 2015) are as follows. An *action* is considered as a causal process that produces a change in the world. The *medium* is the part of the world that undergoes changes because of the action, and also whose state is sensed to incite further actions. A *trace* is the perceivable change made in the medium by an action, which can trigger subsequent actions. A trace that stimulates agents to perform a specific action, i.e. *affords* the action, is called *Affordance*. Affordances typically encode *condition-action* rules, which trigger an agent to perform an action once a certain condition is met. A trace that keeps agents from performing a particular action, is called *Disturbance*.

Heylighen further identifies different variations of stigmergy, depending how the agents interact with the medium (pp. 19 – 27).

This includes the observation that a medium may be worked on by either a single agent, or crowds (*individual vs collective stigmergy*), that agents may take

[2]https://www.w3.org/standards/semanticweb/data

into account either mere existence of certain features in the medium, or quantities of those (*qualitative vs quantitative stigmergy*), that agents may react to direct results of work in their general environment (*sematectonic stigmergy*), or to markers that were deliberately left by other agents (*marker-based stigmergy*), that traces left by agents in the medium can stay until actively being removed by other agents (*persistent traces*) or over time dissipate and vanish (*transient traces*), and finally, that traces in the medium may be observable by every agent working on the medium (*Broadcast*), or only to a limited number of specific agents (*Narrowcast*).

For a very thorough elaboration on the various aspects that we covered here in a very shortened manner, we refer the reader to the original paper (Heylighen, 2015).

4 LINKED DATA AS DIGITAL STIGMERGIC MEDIUM

In nature, the notions of *agent* and *medium* are determined by nature itself: Ants, for example, follow traces of pheromones left by other ants towards lucrative sources of food. Here, ants are the agents, and the ground and food are the medium. Termites use clay as medium, steered by how the shape of their nest (made of this very clay) has already formed. Bees use the air as medium to guide their fellow bees to food sources by dance patterns.

When implementing stigmergy-based algorithms for coordination, *"agents"* are usually considered to be software AI user-agents. These agents operate on a *digital representation* of the to-be-coordinated concepts, which may correspond to real-world physical artifacts (e.g. physical production machines or robots in manufacturing scenarios, cars and traffic lights in traffic). This partition between digital and real world is common in agent-based coordination algorithms (Dipple et al., 2014), with the partitions lately labeled as *Agent Space* for the digital, and *Artifact Space* for the physical representation space (Charpenay et al., 2020).

4.1 Requirements for (Digital) Stigmergic Media

From the notions and variations of stigmergic media in the Section 3.3, we derive the following requirements that a digital medium should fulfill to be suited for use in stigmergy-based systems:

R1 ***(Representation).*** The medium must be able to *represent* entities of the domain in which the coordination algorithm performs, and relations between them. The medium thus serves as *Agent Space*. If artifacts in Artifact Space are target of coordination, the medium must be able to provide a *representation* of the physical entity in the Agent Space, and allow *access* to the physical entity via the mediums, e.g. to switch a real-world traffic light, or start a production process on a production machine.

R2 ***(Accessibility).*** The medium must be *accessible* to the agent in the meaning that an agent must be able to enter the medium to perform actions on it. Furthermore, the agent must be able to navigate through the medium to the point where an action is to be performed.

R3 ***(Observability).*** The medium must be *observable* (readable) for the agent to recognize conditions of condition-action rules to be fulfilled in the medium. For this, the agent needs to be able to at least observe the *existence of effects* (for qualitative stigmergy). The medium should further be suitable to provide:

R3.1 (Interpretability) of observed effects in the medium w.r.t the domain for the agent to correctly set the observed effects into relation with each other.

R3.2 (Quantities): The medium must be able to express *quantities* for coordination by *quantitative* stigmergic effects.

R4 ***(Consistency).*** For *collective stigmergy*, the medium must consistently deliver the same information to different agents at the same point of time. In particular, if an agent induces a change in the medium, following this change, all other agents must observe the changed state as actual state of the medium.

R5 ***(Malleability).*** Agents must be able to *form* and *change* elements in the medium as result of their action. This includes both changing the state of existing entities within the medium (comparable to continuing the construction of a begun nest by a swarm of termites), and add or remove entities from the medium (comparable to leaving pheromone markers, or dissipating markers over time). However, changing the medium should happen in a controlled manner, leading us to the requirement of *Stability*:

R5.1 Stability: Any change to the medium must be possible *without inflicting unwanted effects* to resources outside the scope of a performed action. "Unwanted" is in this case not to be con-

fused with changes that an agent "unintentionally" left as a trace, but to be understood as an effect that changes the state of an entity beyond what was intended by the algorithm.

R6 ***(Scopes).*** The medium must be able to limit visibility of entities and effects in terms of scopes to allow *Narrowcast* of stigmergic effects.

4.2 Linked Data as Stigmergic Medium

We in the following show that read-write Linked Data fulfills all requirements towards a digital stigmergic medium.

Representation is achieved by the notion of representation space of resources (see Sections 3.1 and 3.2): Read-write Linked Data being built around resource oriented architectures provides both the tools and best practices of how to represent both real-world and virtual entities in terms of addressable resources. Connections to physical artifacts is established by having callable HTTP endpoints represented as resources within the medium.

Accessibility is achieved by building Linked Data around HATEOAS principles. Agents can access resources and read information from them by HTTP GET requests. All information needed to interact with a resource is provided by the server that manages the resource. Furthermore, Linked Data defines query interfaces as a common interaction method with linked data graphs. By employing graph query engines like SPARQL[3], agents can identify relevant resources as a result of the queries. Moreover, by following links between related resources, agents may explore Linked Data graphs autonomously. It is not required to host the medium being hosted on a single physical server instance to ensure accessibility. In fact, Linked Data principles state that resolution of URIs should happen transparent, and agents do not need to make assumptions where the actual data is hosted. This allows the medium to be hosted on distributed servers. For queries, SPARQL supports the integration of data from different distributed endpoints via federated queries using the `SERVICE` keyword.[4]

Observability: "Existence" of an effect is manifested in Linked Data by existence of a respective triple pattern in the Linked Data graph. By this, the existence of an effect as precondition for an action can be verified by matching expected triple patterns against the Linked Data graph via SPARQL queries. The statements encoded by triples are moreover *semantically interpretable* by software agents, as commonly established for Linked Data graphs.

Quantities can be modelled in Linked Data graphs either by using respective literal nodes that express a quantity in terms of a fitting datatype directly, or by the number of triples rendered to a respective resource.

Consistency is achieved by the notion of state and representation of resources in a Linked Data system as outlined in Sections 3.1 and 3.2. Access to resources, the represented concept, and modes of interaction are managed by the Linked Data server and provided to clients following HATEOAS principles. By the communication between clients and server being stateless (by following REST and HATEOAS principles), the state of the resource as communicated by the server towards clients is independent of particular clients, and by this, consistent among all clients.

Malleability is achieved by write-capabilites of read-write Linked Data. On resource-level, agents may change the state of a resource by PUT / POST / DELETE requests. On graph level, linked data provides possibilities to change elements in the medium using SPARQL UPDATE requests with INSERT and DELETE statements. The WHERE body of these states moreover allows to take into account preconditions that need to be fulfilled when performing the update.

Stability during updates is achieved by unambiguous identification of relevant resources via IRIs. By default, operations on resources do not have side-effects on other resources, and by this, will not inflict undesired changes to resources other than those that the action was performed on. On RDF graph level, stability during write operations is ensured by that adding triple statements to a resource does not interfere with triples already present: by adding triples, statements about a resource may only become more specific, but never eliminate statements that were present before new triples were added.

Scope can be expressed in read-write Linked Data either by specific triple statements on resources by which agents can filter for specific resources, or by using mechanics of Linked Data datasets and named graphs.[5] Different scopes, i.e. named graphs, are then accessed by agents for example by using `FROM` and `FROM NAMED` clauses in the respective SPARQL queries.

By finding all requirements **R1 – R6** fulfilled by and materialized via concepts of read-write Linked Data systems, we derive that read-write Linked

[3]W3C SPARQL 1.1 Query Language Recommendation (Apr. 2021): https://www.w3.org/TR/sparql11-query/

[4]SPARQL Federated Queries: (Apr.2021): https://www.w3.org/TR/sparql11-federated-query/

[5]https://www.w3.org/TR/rdf-sparql-query/#rdfDataset

Data is without limitations a suitable generic digital medium for stigmergy-based coordination.

5 APPLICATION EXAMPLE

Following, we show how to employ a read-write Linked Data layer as stigmergic medium in Agent Space by an application example from the domain of digital manufacturing. The example is loosely based on the use case presented in (Schraudner and Charpenay, 2020): A (simulated) factory receives orders for simple IoT modules on a ”batch size 1” production line as commonly envisioned in Industry 4.0 (Lasi et al., 2014; Mrugalska and Wyrwicka, 2017). Once an order is received, it is carried out by employing machines which provide the capabilities to perform manufacturing steps necessary for particular steps during the production process, e.g. providing plastic casts for casings, soldering electric circuits, or fixing the final model (see Figure 1).

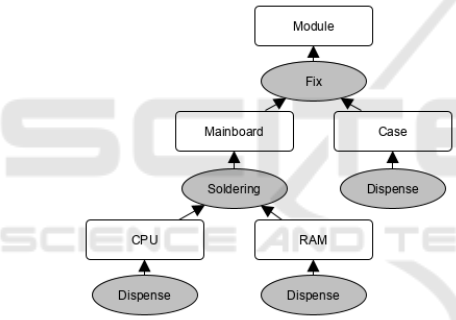

Figure 1: Process of IoT module production used as example.

Orders are executed by AI agents. The need for coordination arises as machines are shared between simultaneously executed orders.

The purpose of the presented coordination algorithm is to find a suitable distribution of machines between agents working on different orders, with the goal to complete each order in the shortest possible time.

5.1 Domain Model

The domain model for our application example is shown in Figure 2: An *order* requests a certain product to be produced. How a product can be assembled is described in *recipes*. These recipes specify what other products are needed as prerequisite, and which manufacturing step is necessary to assemble supply products to a higher level product. Manufacturing steps are provided by units on the shop floor, and can be executed by network interaction endpoints. As common in models for automated production, and also assumed in (Charpenay et al., 2020) and (Schraudner and Charpenay, 2020), we assume production units to have callable network endpoints by which their particular production step can be executed. Affordance- and disturbance markers are used to encourage or discourage the use of a certain interaction endpoint.

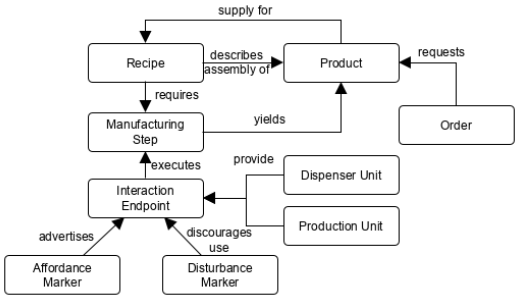

Figure 2: Domain model of the chosen application example.

Listing 1: Example of a production recipe using schema and steps vocabularies.

```
recipes:main-module rdf:type schema:HowTo ;
  schema:about mainboard:product ;
  schema:step steps:solder ;
  schema:supply [
    rdf:type schema:HowToSupply ;
    schema:item cpu:product ] ,
  [ rdf:type schema:HowToSupply ;
    schema:item ram:product ] .
```

A recipe specifies the produced artifact by the `schema:about` predicate, the required supply from which the product is created (indicated the `schema:supply` predicate), and the production step that needs to be performed to combine the specified supplies to the resulting product via the `schema:step` predicate (see Listing 1). `schema:` denotes the namespace of the schema.org ontology [6]. We assume a set of supply materials to be provided to the factory without the need for specific production. These supply materials will be provided by dispenser units, and do not require any additional supplies.

Products are described in terms of an RDF class, e.g. `(<#product>, rdf:type, cpu:product)`.

Dispenser and production units specify their (callable) execution endpoint as `td:InteractionPattern` in a set of triples that is referenced via `td:providesInteractionPattern`. The Interaction Pattern specifies the step carried out by the respective unit (see also Listing 2). Dispensers

[6] https://schema.org/

refer to the class of dispensed products via a triple (<#unit>, schema:yield, <#productClass>), with <#productClass> referring to the RDF class of the produced product. Dispenser units can dispense more than one class of products. Production units do not specify particular products that are produced at the unit to allow for various products that are produced with the same step to be produced at the same machine. Instead, they provide information about the type of provided production step via a triple (<#unit>, schema:step, steps:<type>). An example of a simple soldering unit is shown in Listing 2.

Interaction Patterns on machines describe particular *actions* that agents may perform by to trigger the execution of the respective production step on the physical machine. This is done by resolving the URI that is provided by the respective resource, and that is identified by the property path td:isAccessibleThrough/td:href, with td: denoting the namespace of the Web Thing Description ontology [7].

Listing 2: Example of a simple description of a workstation that performs a soldering step. The soldering action is executed by calling the respective referenced URI.

```
sol:station-1 a td:Thing ;
  td:thingName "Soldering Station 1"^^xsd:
      string ;
  td:providesInteractionPattern sol:soldering .

sol:soldering a td:InteractionPattern ;
  td:interactionName "solder"^^xsd:string ;
  schema:step steps:solder ;
  td:isAccessibleThrough [
    td:href <http://10.2.100.17/solder/>
  ] .
```

5.1.1 Affordances and Disturbances

Listing 3: Example of an affordance marker resource that advertises a steps:soldering interaction as relevant for the current order.

```
<urn:uuid:a526>
    a stigmergy:marker lef ;
    stigmergy:marked  sol:soldering ;
    stigmergy:scope  order:module ;
    schema:supply  cpu:product,
                   ram:product ;
    schema:yield  mm:product .
```

Affordances will advertise td:InteractionPattern resources as callable endpoint to some executing agent. Affordances are *markers* that are left on a td:InteractionPattern.

Listing 3 shows an example of such a marker: The marker gives information about which Interaction Pattern it marked (via stigmergy:marked), for which order the respective pattern needs to be executed (via stigmergy:scope), whether or not the respective step needs particular supplies to be present to be executed (schema:supply), and finally, which product will be the result of calling the respective InteractionPattern resource (schema:yield). A marker can link to one or more interaction patterns. If more than one interaction pattern is marked, it is up to an executing agent to choose which of the endpoints to call.

Disturbance markers will discourage agents from visiting a marked resource. If an affordance marker links to several resource endpoints, an executing agent will decide for an endpoint that is the least influenced by disturbance markers. The complete algorithm will be detailed out in Section 6.

[7] https://www.w3.org/2019/wot/td

6 EXAMPLE ALGORITHM

We now show how to realize a stigmergy-based coordination algorithm based on the definition of the Linked Data layer from the previous section.

The algorithm to execute a coordinated production process for multiple orders is implemented in two steps by two classes of agents. We divide agents into *marker* agents and *builder* agents. Marker agents traverse graphs in the agent space and generate *production markers* as affordances on resources in the Artifact space as shown in Listing 3. Builder agents are attracted towards the respective endpoints by the affordances left by the marker agents and execute those production endpoints that were marked in the scope of the current order, given that the production requirements (supplies) are met.

6.1 Marker Agents

The goal of a marker agent to identify all *suitable* production units that will be involved in the process of producing a particular order. For this, marker agents will traverse recipe resources and leave markers on resources as follows:

The agent maintains a list unvisited of nodes it would like to visit, but has not yet.

1. Check for *order* resources that have not yet been handled, i.e., do not carry a mark. Follow the link via the schema:orderedItem property to the resource that represents the class of the ordered product and add it to unvisited. Mark the order as *handled*.
2. From a resource *r* in unvisited, find a respective recipe blueprint *b* that contains a triple (*b*,

schema:about, *r*), i.e., the recipe for the respective product.

3. Check for a schema:step link, and *visit all interaction patterns i* matching the schema:step; if the step is steps:dispense, find the respective interaction patterns of dispenser units that schema:yield *r* .
4. Leave a *mark* on each visited *i* (for both production and dispenser, cf. Listing 3).
5. For each resource *s* in schema:supplies of *b*, add *s* to unvisited. If no schema:supply is specified, or resource points to an empty set (rdf:nil), do nothing. Remove the current resource *r* from unvisited.
6. If unvisited is empty, **terminate**; else, **go to 2**.

The mark which is left by the agent in Step 4. follows the structure of the example shown in Listing 3. It includes information about the order in the scope of which it was placed, and will moreover specify the required supplies *s* for this step. Markers may be scoped by order, using a triple (*marker*, stig:scope, *order*) on the marker resource (*Narrowcast*), or unscoped and by this visible for every other agent (*Broadcast*). Following the given algorithm, a marker agent is solely driven by the structure of the knowledge graph that is formed between product and blueprint descriptions. Each subsequent step is solely decided by the state of the currently visited resource. Its behavior can by this be classified as a *sematectonic stigmergic* agent.

6.2 Builder Agents

Builder agents are attracted to markers left by marker agents and call the respective InteractionPattern endpoints. A Builder agent for this proceeds in the following manner:

1. Scan for all markers *m* left by marker agents. If the builder agent is bound to a specific scope (i.e. fulfilling a particular order), it will only follow markers in its scope (i.e., with a matching (*m*, stig:scope, *order*) triple present.
2. For each *m*, the checks, e.g. via a fitting SPARQL query, if for each *supply s* specified by the marker via (*m*, schema:supply, *s*), there exists a product *p* that is a product of class *s*, as encoded by a triple (*p*, rdf:type, *s*).
3. For each *m* for which supplies are fulfilled, the visit the InteractionPattern resource *i* that is marked via (*m*, stig:marked, *i*) and that carries the *least amount of disturbance markers*. Execute the action endpoint that is identified via td:isAccessibleThrough/td:href.
4. Leave a disturbance marker on the interaction pattern resource, and removes the affordance marker.

By following a marker trace left by other agents, we employ a *marker-based* stigmergy approach. The supplies specified on the marker, in combination with a reference to callable resources for execution of the respective interaction step, encode a condition-action-rule that the agent follows.

6.3 Correctness of the Algorithm

An order specificies the expected result of one instance of the algorithm, namely the specific product that is to be produced in the end. By marker agents starting from the expected goal (the ordered product), and following links backwards through the needed supplies, it is ensured that over the total production process, all needed supplies will be available eventually. Marker agents do not need to keep memory of the goal they are following, but are guided entirely by the structure of the Linked Data medium. It can be easily shown that the marker agent's algorithm will terminate as soon as all dispensable supplies (leaf nodes in the graph in Figure 1) are provided with a marker. The builder agent's algorithm will terminate when the last marker is consumed. Builder agents are not restricted to follow only markers of a specific order. Consuming a marker and triggering the respective action will always result in a product that was previously found to be a requirement by some marker agent. Eventually, by builder agents executing endpoints for products with rising complexity as supplies are more and more met, the ordered product will be produced.

If several orders are executed in parallel, production units (and dispensers) will simply receive several independent markers. By having separate markers per order, and having builder agents removing the marker they followed after executing the production step, it is ensured that for every order, every production step is executed exactly once. The concept may be extended for products to require more than one instance of a supply product. In this case, a marker agent would leave a marker per required instance of a supply.

The opportunity for coordination arises in Step 3. of the builder agent algorithm: For every recipe, markers are left on every machine that is capable of carrying out the needed production step. When following the marker trace, builder agents have a choice which of the marked machines they actually execute.

The decision for which machine to call for to execute the step is based on the number of disturbance markers left on the resource: The more agents visit, means, the busier the machine already is with executing orders, the more disturbance markers are left on the machine, and agents will be more likely to divert to other machines to complete their order.

The algorithm at this point ignores transport of products on the shop floor. A more sophisticated heuristic may take into account also transport times between machines between the different steps.

6.4 Implementation

We implemented the example using the Unity 3D game engine to simulate the factory, a Fuseki triple store to host the read-write Linked Data medium, and the AJAN agent platform[8,9] (Antakli et al., 2019) to implement the behaviors of both marker and builder agents. All related resources will be published on GitHub:
https://github.com/BMBF-MOSAIK/stigmergy-demo

7 DISCUSSION

In the following, we analyse the use of Linked Data as stigmergic medium based on findings from the algorithm w.r.t. benefits of stigmergic systems (see also (Heylighen, 2015, pp.13-14)):

Agents do *not plan or anticipate*, but only follow links between resources in the Linked Data medium. The condition-action-rules that determine which resources to visit are generic. Agents do not need to make per-resource decision whether following the rule is beneficial for the goal, or not. The "goal" (production of a specific order) is, in particular, not known to an agent, and reaching any goal is no condition for termination of the algorithm.

Memory-less agents are sufficient by storing all relevant information that arises during execution in terms of resources in the medium. Same goes for *communication between agents*, which is eliminated by limiting interaction to following markers left by other agents. Agents are moreover not *aware* of each other, as their interaction is limited entirely to the Linked Data medium. This also implies that agents *do not need to be simultaneously present*.

The correct *sequence* of steps arises naturally by including information about required supplies, both when a marker agent decides which recipe resources to visit next, and when a builder agent decides which marked production resource to visit next. There is no requirement to explicitly model (and by this, impose) sequences during execution of a particular production in a form like *"after you have successfully visited this resource, continue with that specific resource over there"*. Instead, sequence arises implicitly from the condition-action rules encoded in the Linked Data medium.

Non-necessity for commitment is achieved by having no explicit assignment of tasks to agents, but have agents decide which resource to visit, and how to interact with it (e.g., perform their competent action), solely on the state of *resources in the medium*. Any agent can pick up any task at any point in time according to the agents' competence.

Finally, as it is obvious from the algorithms of both marker and builder agents, that there is *no centralized coordination or control* authority that agents need to consult, or by which they are controlled. Coordination arises solely from resource states and markers left in the medium, as already discussed in Section 6.3.

8 CONCLUSION AND FUTURE WORK

In this paper, we have thoroughly analyzed read-write Linked Data as a digital medium for stigmergy-based coordination mechanisms. We have based this analysis on common general characteristics of stigmergic systems in literature. By identifying direct correspondences between these and central features of Linked Data systems, we finally achieved to show that read-write Linked Data provides a perfect digital medium for stigmergy-based coordination algorithms. Finally, we provided an example of how to employ these findings in a coordination algorithm for customized digital production.

Currently, the presented algorithm is to be considered a basic conceptual example to show that decentralized stigmergy-based coordination works. It is not yet intended to show improved efficiency and scalability over existing approaches. We do plan to further improve and evaluate the algorithm with focus on these aspects in future work. We moreover plan to demonstrate the applicability of Linked Data as a stigmergic medium by more complex examples from different domains.

For actual application in a production setting, the algorithm as presented omits elements like location of products and production units on the shop floor,

[8] https://github.com/aantakli/AJAN-service
[9] https://github.com/aantakli/AJAN-editor

and by this, necessary steps and tools for transport of products. We are working on implementing these concepts in an extended version of the algorithm.

ACKNOWLEDGEMENTS

This work has been supported by the German Federal Ministry for Education and Research (BMBF) as part of the MOSAIK project (grant no. 01IS18070-C).

REFERENCES

Alfeo, A. L., Cimino, M. G., Egidi, S., Lepri, B., and Vaglini, G. (2018). A Stigmergy-Based Analysis of City Hotspots to Discover Trends and Anomalies in Urban Transportation Usage. *IEEE Transactions on Intelligent Transportation Systems*, 19(7):2258–2267.

Antakli, A., Spieldenner, T., Köster, M., Groß, J., Herrmann, E., Rubinstein, D., Spieldenner, D., and Zinnikus, I. (2019). Optimized coordination and simulation for industrial human robot collaborations. In *International Conference on Web Information Systems and Technologies*, pages 44–68. Springer.

Berners-Lee, T., Hendler, J., Lassila, O., et al. (2001). The semantic web. *Scientific american*, 284(5):28–37.

Bizer, C., Heath, T., Idehen, K., and Berners-Lee, T. (2008). Linked data on the web (ldow2008). In *Proceedings of the 17th international conference on World Wide Web*, pages 1265–1266.

Bonabeau, E., Henaux, F., Guérin, S., Snyers, D., Kuntz, P., and Theraulaz, G. (1998). Routing in telecommunications networks with ant-like agents. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 1437(1):60–71.

Charpenay, V., Schraudner, D., Seidelmann, T., Spieldenner, T., Weise, J., Schubotz, R., Mostaghim, S., and Harth, A. (2020). Mosaik: A formal model for self-organizing manufacturing systems. *IEEE Pervasive Computing*.

Chiong, R. (2009). *Nature-inspired algorithms for optimisation*, volume 193. Springer.

Chopra, S., Notarstefano, G., Rice, M., and Egerstedt, M. (2017). A Distributed Version of the Hungarian Method for Multirobot Assignment. *IEEE Transactions on Robotics*, 33(4):932–947.

Ciancarini, P., Gorrieri, R., and Zavattaro, G. (1997). Towards a calculus for generative communication. In *Formal Methods for Open Object-Based Distributed Systems*, pages 283–297. Springer.

Cicirello, V. A. and Smith, S. F. (2004). Wasp-like agents for distributed factory coordination. *Autonomous Agents and Multi-Agent Systems*, 8(3):237–266.

De Nicola, R., Di Stefano, L., and Inverso, O. (2020). Multi-agent systems with virtual stigmergy. *Science of Computer Programming*, 187:102345.

Dipple, A., Raymond, K., and Docherty, M. (2013). Stigmergy within Web Modelling Languages : Positive Feedback Mechanisms. *eprints.qut.edu.au*.

Dipple, A., Raymond, K., and Docherty, M. (2014). General Theory of Stigmergy: Modelling Stigma Semantics. *Elsevier*.

Dipple, A. C. (2011). Standing on the shoulders of ants: Stigmergy in the web. In *Proceedings of the 20th international conference companion on World wide web*, pages 355–360.

Dorigo, M. and Blum, C. (2005). Ant colony optimization theory: A survey. *Information Sciences*.

Fielding, R. T. and Taylor, R. N. (2000). *Architectural styles and the design of network-based software architectures*, volume 7. University of California, Irvine Irvine.

Gelernter, D. (1985). Generative communication in linda. *ACM Transactions on Programming Languages and Systems (TOPLAS)*, 7(1):80–112.

Heylighen, F. (2006). Mediator evolution: a general scenario for the origin of dynamical hierarchies. *Worldviews, Science and Us.(Singapore: World Scientific)*, 44:45–48.

Heylighen, F. (2015). Stigmergy as a universal coordination mechanism: components, varieties and applications. *Human Stigmergy: Theoretical Developments and New Applications; Springer: New York, NY, USA*.

Jevtić, A., Gutierrez, Á., Andina, D., and Jamshidi, M. (2012). Distributed bees algorithm for task allocation in swarm of robots. *IEEE Systems Journal*, 6(2):296–304.

Kanamori, R., Takahashi, J., and Ito, T. (2014). Evaluation of traffic management strategies with anticipatory stigmergy. *Journal of Information Processing*, 22(2):228–234.

Korošec, P., Šilc, J., and Filipič, B. (2012). The differential ant-stigmergy algorithm. *Information Sciences*.

Krieger, M. J., Billeter, J. B., and Keller, L. (2000). Ant-like task allocation and recruitment in cooperative robots. *Nature*, 406(6799):992–995.

Lasi, H., Fettke, P., Kemper, H.-G., Feld, T., and Hoffmann, M. (2014). Industry 4.0. *Business & information systems engineering*, 6(4):239–242.

Lassila, O., Swick, R. R., et al. (1998). Resource description framework (rdf) model and syntax specification.

Lucchi, R., Millot, M., and Elfers, C. (2008). Resource Oriented Architecture and REST. *Assessment of impact and advantages on INSPIRE, Ispra: European Communities*.

Matarić, M. J., Sukhatme, G. S., and Østergaard, E. H. (2003). Multi-robot task allocation in uncertain environments. *Autonomous Robots*, 14(2-3):255–263.

Mayer, S., Verborgh, R., Kovatsch, M., and Mattern, F. (2016). Smart configuration of smart environments. *IEEE Transactions on Automation Science and Engineering*, 13(3):1247–1255.

Mrugalska, B. and Wyrwicka, M. K. (2017). Towards lean production in industry 4.0. *Procedia engineering*, 182:466–473.

Privat, G. (2012). Phenotropic and stigmergic webs: The new reach of networks. *Universal Access in the Information Society*, 11(3):323–335.

Ricci, A., Omicini, A., Viroli, M., Gardelli, L., and Oliva, E. (2007). Cognitive stigmergy: Towards a framework based on agents and artifacts. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 4389 LNAI:124–140.

Schraudner, D. and Charpenay, V. (2020). An HTTP/RDF-Based Agent Infrastructure for Manufacturing Using Stigmergy. (01):197–202.

Tzanetos, A., Fister Jr, I., and Dounias, G. (2020). A comprehensive database of nature-inspired algorithms. *Data in Brief*, 31:105792.

Valckenaers, P., Hadeli, Germain, B. S., Verstraete, P., and Van Brussel, H. (2007). MAS coordination and control based on stigmergy. *Computers in Industry*, 58(7):621–629.

Verborgh, R., Steiner, T., Deursen, D. V., d. Walle, R. V., and Vallés, J. G. (2011). Efficient runtime service discovery and consumption with hyperlinked restdesc. In *2011 7th International Conference on Next Generation Web Services Practices*, pages 373–379.

Yu, X. and Cheng, T. (2020). Research on a Stigmergy-driven & MAS-based Method of Modeling Intelligent System. pages 1042–1047.