Proceedings of the 2022 Winter Simulation Conference
B. Feng, G. Pedrielli, Y. Peng, S. Shashaani, E. Song, C.G. Corlu, L.H. Lee, E.P. Chew, T. Roeder, and P. Lendermann, eds.

GRAPH REPRESENTATION AND EMBEDDING FOR SEMICONDUCTOR MANUFACTURING FAB STATES

Benedikt Schulz*
Christoph Jacobi*

Karlsruhe Institute of Technology (KIT)
Institute for Material Handling and Logistics (IFL)
Gotthard-Franz-Straße 8
76131 Karlsruhe, GERMANY

Andrej Gisbrecht
Angelidis Evangelos

Robert Bosch GmbH
Tübinger Straße 123
72762 Reutlingen, GERMANY

Chew Wye Chan
Boon Ping Gan

D-SIMLAB Technologies Pte Ltd
8 Jurong Town Hall Road
Singapore 609434, SINGAPORE

ABSTRACT

Due to the enormous complexity of semiconductor manufacturing processes, tasks like performance analysis, forecasting, and production planning and control necessitate detailed knowledge about the current state of the manufacturing system. Usually, models and methods for these tasks incorporate feature selection and engineering to extract relevant feature sets from the vast number of available features. However, sets of independent features may not retain structural information that captures interdependencies between entities. To address this challenge, a graph representation model for semiconductor manufacturing fabs that captures structural information, such as the interdependencies of machines, lots, and routes, is presented. The model comprises the essential procedures in semiconductor manufacturing processes, namely process flows, material transfer, setup, and maintenance activities. Finally, we use representation learning to embed graph snapshots into a low-dimensional space. These embeddings can serve as input for a scheduling engine or a performance analysis tool.

1 INTRODUCTION

Semiconductor manufacturing processes are among the most complex industrial processes. Due to this complexity, performance analyses and forecasts, as well as production planning and control are non-trivial tasks which require detailed knowledge about the state of the manufacturing system at any given point in time. The state of the manufacturing system is described by the set of all features and entities in the system as well as their interdependencies. Many approaches for performance analysis or production planning extract extensive feature sets to describe the state of the manufacturing system. However, extracting meaningful feature sets from a real data base is computationally expensive (Schelthoff et al. 2022), and sparse feature sets usually do not allow to describe the state comprehensively, as they fail to adequately represent the interdependencies between the features and entities. Therefore, we propose to use structural data instead of independent features to represent meaningful fab states. Based on experience in developing simulation models for the semiconductor industry and a broad understanding of the various parameters, we identify those that need to be integrated into a fab state model. Laipple et al. (2018) present a generic data model for semiconductor supply chains that covers many parameters that are important in semiconductor manufacturing.

In this paper, we aim to enrich the generic data model with the interdependencies in a semiconductor fab. In Section 2, we present the state of the art for simulation of semiconductor manufacturing systems and approaches for fab state representation. Section 3 introduces a graph representation model for semiconductor fab states. In Section 4, we propose an approach to map graph snapshots into a low-dimensional space, where the embeddings are intended to be used for subsequent machine learning algorithms as suggested by Narayanan et al. (2017). Section 5 concludes the paper.

2 STATE OF THE ART

2.1 Simulation of Semiconductor Manufacturing Systems

When simulating production and logistics systems, discrete-event simulation is a popular approach that allows to take stochastic and dynamic effects into account (Mönch et al. 2013). Here, the system state changes at certain points in time that are associated with an event. Each system state can be used in an encoded format as input for e.g. production planning and scheduling algorithms. Such system states can be generated either offline based on logged data from simulation runs or online while a simulation is running. Granularity and fidelity of the simulation prescribe the scope for production planning and scheduling decisions that can be taken based on system states extracted from the simulation. In particular, a simulation might be focused on a short-, medium- or long-term perspective.

Discrete-event simulation has been widely adopted in the wafer fabrication industry for the purpose of WIP flow forecasting to identify daily operation bottlenecks (Seidel et al. 2017), dynamic capacity planning to determine product mixes for the best cycle time and line stability (Nyhuis and Alves (2002), Zhang et al. (2008), Geng and Jiang (2009), and Seidel et al. (2019)), workload forecasting for preventive maintenance scheduling (Scholl et al. 2012), and lot level forecast to predict expected delivery date to customer (Scholl et al. (2011), Scholl et al. (2016), and Seidel et al. (2020)). Typical modeling elements required for a wafer fab simulation model are summarized in Tables 1 and 2.

The use cases outlined earlier were realized with different levels of modeling fidelity. For example, WIP forecasting at daily granularity for each work center is required by Seidel et al. (2017) and Scholl et al. (2012). In these cases, the simulation models were built considering detailed information such as short term equipment level dedication blocking, equipment throughput as a function of available chambers, complete process flows for production lots, global and local dispatching rules at each equipment type, and

Table 1: Model category *Production*.

Model element	Description
Process flow	Sequence of steps (with associated step attributes such as sampling rate, hold rate, rework rate, or split rate) products pass from wafer start to fab out.
Time window	Minimum and maximum time between two steps of a process flow.
Equipment configuration	Tool type (batch, lot, single wafer, or wafer batch), chambers, number of load ports, run model (serial/parallel).
Equipment dedication	Equipment capability that defines which lot can be processed on an equipment (by recipe, combination of recipe and product, or combination of recipe, product and stage).
Equipment down	Statistical distribution for mean-time-to-failure and mean-time-to-repair to model equipment downtime.
Reticle	Additional resource required for lot processing at lithography tools.

Table 2: Model category *Snapshot* (warm start simulation).

Model element	**Description**
WIP (Work-In-Progress)	Lots in the fab at a specific point in time and their associated attributes such as current process flow, current step, in-process/in-queue/transit, priority, number of wafers, start date, delivery due date, and hold.
Wafer Start	Lots to be started in the production line during the simulation horizon, provided at lot level.
Equipment dedication	Temporary dedication blocking by recipe, product, stage, layer, and combinations of different lot attributes.
Equipment/chamber initial state	State (productive, standby, scheduled down, unscheduled down) of the equipment/chamber at a specific point in time.

random events (such as rework, hold and equipment down) derived from a short term time horizon (not more than three months). In contrast, the simulation model built for the dynamic capacity planning use case is at a higher level of abstraction (Seidel et al. 2019). Mosinski et al. (2017) discuss an approach to create an abstracted simulation model without losing model accuracy for the purpose of dynamic capacity planning.

To reflect an accurate effect of scheduling and dispatching decisions, a high fidelity simulation model like those presented by Seidel et al. (2017) and Scholl et al. (2012) is required. In addition, a simulation platform that enables the generation of training and test data for our modeling approach is required. This simulation platform has been built with the D-SIMCON Simulator (D-SIMLAB 2022) and provides the following categories of functionalities:

- Execution control: The simulation platform is able to start, pause, branch, and rollback the simulation run. Branching is a feature where a new simulation instance can be started from a point in time and run independently of other simulation instances. Rollback of a simulation run is supported by restoring the simulation state at an earlier point in time where a snapshot was taken.
- Command (input): The simulation platform is able to feed lot dispatching decisions to the simulation model, that is then executed for a time horizon determined by the simulation platform before performance measures are taken.
- Effect/State (output): The simulation platform is able to retrieve the performance impact of decisions from the simulation model. The performance impact can be measured at fab, product, work center, equipment, or lot level. Some examples of these performance measures are: cycle time, on-time delivery, utilization, and throughput. The state of the system in the simulation model is also essential in driving the decisions made by a scheduling engine. Some features of system states are queuing lots in front of a work center and their associated attributes, equipment states (busy, idle, or down), number of lots for products, etc.

2.2 Approaches for Fab State Representation

As production planning decisions depend on the current state of a fab, this fab state has to be represented in the input data for production planning algorithms. In industry, data regarding the fab is usually stored in databases, e.g. the Manufacturing Execution System (MES) and Enterprise Resource Planning (ERP) Systems (Mönch et al. 2013). Based on the fact that data storage and modeling is usually done by each organization independently and thus there are limited opportunities to synchronize optimization efforts, Laipple et al. (2018) propose a generic data model to standardize the description of data in semiconductor manufacturing supply chains. This generic data model also covers data on the shop-floor level which is a relevant input for production planning algorithms. While these systems and concepts provide the required raw data, it is important to preprocess the data to get a format that is appropriate to apply production planning algorithms (Mönch et al. 2013). Approaches for production planning are usually based on some schema to model the required input data. Models and methods for data representation are the base for successful production planning applications on the other hand (Chien et al. 2011).

Many approaches for production planning or performance analysis utilize a selection of parameters out of the huge set of available parameters that in total represent the current state of a system. Sarin et al. (2011) give an overview of dispatching rules and the parameters used respectively. Parameters that are commonly used by production planning and scheduling algorithms are for example due date, processing time, utilization and queue length. In some studies, the considered parameters are selected dynamically. Li et al. (2013) present an adaptive dispatching rule that takes into account real-time information to achieve better operational performance. Kim et al. (2020) propose a prediction method for a machine allocation problem of production scheduling taking into account constraints from the automated material handling system as well. With regard to the selection of appropriate parameters, some studies are concerned with the approach to select some parameters and withdraw others. Thus, they intend to represent or at least approximate the state of a system by a limited set of parameters. Schelthoff et al. (2022) use real operational data to investigate influencing features for waiting time estimation of operations in high product-mix / low-volume semiconductor manufacturing fabs and present a framework for feature selection. Qiao et al. (2013) investigate which parameters play key roles in manufacturing scheduling according to a specific performance criterion using a genetic algorithm based selection approach.

An approach to store data associated with a production system in a structured way are so-called expert or knowledge-based systems. They are composed of a knowledge base where the knowledge of human experts is stored in a formalized way and an inference engine that operates on the knowledge base to solve a problem at hand (Gupta and Sivakumar 2006). De and Lee (1998) present a knowledge-based scheduling system for semiconductor testing with a knowledge base that is composed of environmental, procedural and structural knowledge. Procedural and structural knowledge refers to rules, e.g. for conflict resolution, and information about the solution process respectively. Environmental knowledge captures information about the environment like process flows and the current work in process. In particular, De and Lee (1998) define nine entities like workstations and operations that are associated with different aspects that are relevant for planning purposes. Additionally, they define relationships among the entities that specify one-to-may-relations.

Another formal approach to describe and analyze complex discrete-event systems like in semiconductor manufacturing is Petri Nets (Gupta and Sivakumar 2006). The domain of Petri Nets provides various concepts that allow to model semiconductor manufacturing systems, e.g. priorities, time delays as well as dedicated and shared resources (Zhou and Jeng 1998). Consequently, a snapshot of a Petri Net can be interpreted as a representation of the current state of the modeled system. However, the complexity of Petri Nets increases drastically with the number of reachable states and events. Lee and Favrel (1985) present a hierarchical reduction method that allows for a hierarchical decomposition of large systems into multilevel subsystems and thus a step-wise analysis and easy verification of Petri Nets.

Decomposition approaches are a common way to handle scheduling problems that are characterized by high complexity. Therefore, one has to decompose the problem into appropriate sub-problems, model the interactions between the sub-problems, identify and apply solution procedures to the sub-problems and merge the solutions of the sub-problems in the end (Ovacik and Uzsoy 1997). A popular approach to model interactions between sub-problems is the disjunctive graph representation. The corresponding graph is composed of nodes each representing an operation of a job on an equipment, directed edges each representing a relation between nodes, for example a process flow or a scheduling decision, and undirected edges between nodes representing an operation on equipments of the same equipment group. Additionally, weights representing processing times can be assigned to directed edges. There are various extensions that allow to model specific characteristics of semiconductor manufacturing systems like parallel machines, sequence-dependent setup times, batching and re-entrant process flows (Mönch et al. 2013). Based on disjunctive graph representations, production schedules can be generated using algorithms for path-finding. In order to handle systems where the state changes dynamically over time, this can be done frequently based on a rolling horizon (see e.g. Drießel and Mönch (2012)). Thus, a disjunctive graph representation provides a structure to generate production schedules but is not intended to represent the current state of a system.

3 GRAPH REPRESENTATION OF THE GENERIC DATA MODEL

Formulating real world data as a graph is very convenient to represent complex relationships among entities in the data (Mahdavi et al. 2020). Graph representations have been proven to be useful in several research areas like in the analysis of social networks (Hoff et al. (2002) and Bhagat et al. (2011)), statistical relational learning (Nickel et al. 2016), and predicting a multi-cellular function (Zitnik and Leskovec 2017).

In this section, we present an approach to transfer the generic data model for semiconductor manufacturing supply chains of Laipple et al. (2018) into a graph representation model. In contrast to the concept of knowledge graphs, where current research focuses on knowledge graph completion and predicting potential relationships between entities (Chen et al. 2020) as well as knowledge reasoning that aims at deducing new judgments (conclusions) from one or several existing premises (Chen, Jia, and Xiang 2020), the graph model presented in the following is solely an encoding formalism that allows to model structural information like the relation of different parameters.

3.1 Generic Data Model for Semiconductor Manufacturing Supply Chains

The generic data model for semiconductor manufacturing supply chains developed by Laipple et al. (2018) allows to model different aggregation levels of semiconductor manufacturing data. It is composed of master and tracing entities. Master entities describe the system itself, for example, fabs, equipment, setup, maintenance, material transfer and operations. Tracing entities refer to the master entities via foreign key relations and store all time dependent system changes like lot states, lot events and equipment states. Therefore, tracing entities cover dynamic effects based on the static system representation with master entities.

3.2 Graph Representation

In this section, we transfer the generic data model of Laipple et al. (2018) into a graph representation. We introduce the essential mechanisms of our model by zooming in on a small fragment of a production system with four machines (cf. Figure 1). The graph snapshots introduced in the following can be interpreted as fundamental building blocks that can be combined to model manufacturing processes of larger portions of a fab, a plant, or a supply chain. Our focus is on modeling the essential procedures in the semiconductor manufacturing industry. Therefore, we first introduce a graph representation to model processing of lots on machines, and then present graph representations for material transfer between machines as well as setup and maintenance activities.

Figure 1 depicts the different types of nodes and edges of the graph representation for a situation, where lot 1 is currently in operation 1 at machine 2 (production equipment, Eqp_2) with processing time d_1. Machines Eqp_1 and Eqp_2 provide similar processes and are therefore both associated with equipment group $EqpG_1$ (the same applies for machines Eqp_3 and Eqp_4 in $EqpG_2$). Each equipment group is associated with a production area and the technology used (that is, lithography, diffusion, plasma etching, etc.). The machines can perform a set of operations $Op_1, ..., Op_n$, which correspond to one production step, respectively. The sequence of operations a lot passes through builds a route; the sequence of routes is determined by the product type of each lot. An important feature of the graph representation is time. We model periods of time (e.g. for processing, transport, maintenance, etc.) by means of a discrete set of duration nodes, $d_1, ..., d_n$. We propose to associate each duration node with a specific value and to mark periods of the same length with the same node and periods of different lengths with different nodes. The

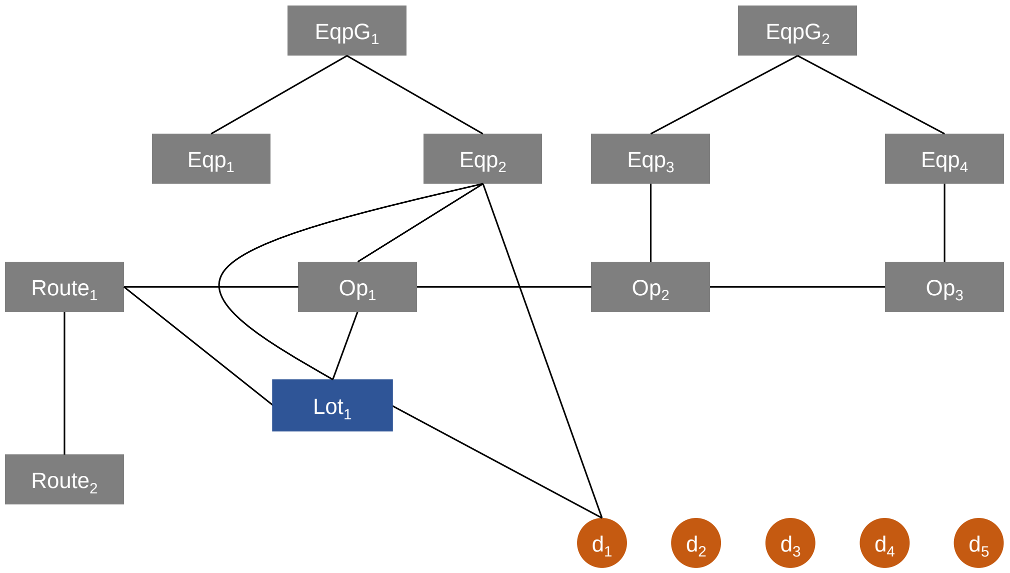


Figure 1: Snapshot of process flow with lot 1 in the graph model.

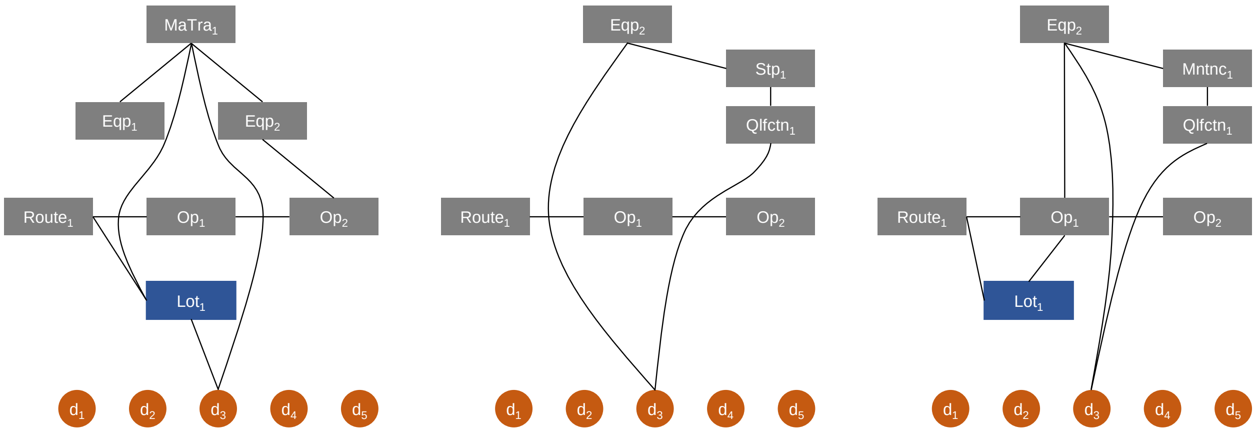
entities equipment group, machine, operation, route, and product level (i.e. lot) are formalized by Laipple et al. (2018) as so-called master entities.

The edges in the graph represent affiliations in the hierarchy of entities, as well as transactions like product flows. Not shown in Figure 1, but conceivable for modeling larger networks are higher hierarchical levels of the production network (e.g. fab, plant, supply chain), where equipment groups are combined. Edges on this level can be considered as being static since equipment groups do not dynamically dissolve and reform and routes are maintained during the production cycle. However, new machines and new products with previously unused routes may be introduced to the fab. In this case, new nodes and edges have to be added to the graph. As this is expected to happen rather occasionally, we assume the edges to be static.

In contrast, we consider an edge to be dynamic if it exists only briefly over time or if start and end nodes change frequently. Naturally, edges connected with nodes representing lots are dynamic. In the snapshot in Figure 1, lot 1 is currently on route 1. The edge between the two nodes is maintained until the last operation in route 1 is completed and the next operation in route 2 begins. It is important to keep track of the route for each lot since the same operation may be executed on different routes (at different stages of production). In Figure 1, lot 1 is currently in Op_1 on Eqp_2, which lasts as specified by d_1. Note that the dynamic edges to connect the entities machine, lot and duration form a cycle. This is important in order to ensure an unambiguous allocation of these entities when the same operation is carried out simultaneously by several machines.

Beyond the graph representation of process flows, we developed graph representations of non-value adding activities. First, we consider material transfer between machines (cf. Figure 2a). In semiconductor wafer fabs, material transfer is usually performed by an Automated Material Handling System (AMHS), but other transfer modes such as AGV, milkrun, and manual are also common. Therefore, we introduce a generic material transfer node $MaTra_1$ between machines which is formalized as master entity in the generic data model of Laipple et al. (2018). On the lines of process flows, the entities material transfer, lot and duration are connected in a cycle in order to ensure an unambiguous allocation of these entities in the case of simultaneous transportation of several lots.

From a modeling perspective, setup (cf. Figure 2b) and maintenance (cf. Figure 2c) activities are very similar. In contrast to the previously discussed material transfer, setup and maintenance activities require specialized qualification, that is, a worker with specific technical and operational experience and training



(a) Material transfer (b) Setup (c) Maintenance

Figure 2: Snapshots of non-value adding activities in the graph model.

must engage in the activity. Setup, maintenance and worker qualification are formalized as master entities in the generic data model (Laipple et al. 2018). Obviously, worker qualification is limited and thus, following the approach for modeling time, we assume a limited number of qualification nodes $Qlfctn_1,\ldots,Qlfctn_n$ in the graph. Executing setup and maintenance tasks is modeled by a cycle of dynamic edges that connect the entities machine, duration, qualification and setup (maintenance). Finally, after setup and maintenance activities are completed, processing can resume. Note that, by definition, setup changes the operation to be performed by the machine. In contrast, maintenance activities may interrupt processing a lot and processing of the lot resumes after maintenance was completed. This will, however, vary according to the situation, and a change of the lot type after maintenance is also conceivable.

4 REPRESENTATION LEARNING FOR DIMENSIONALITY REDUCTION OF FAB STATES

Graph representations in general and the graph representation approach introduced in Section 3 can be computationally expensive, for example when being applied on fab or even supply chain level in the semiconductor manufacturing domain. In the field of research on representation learning, different approaches to learn to encode graph structures with embeddings of lower dimensionality are proposed, e.g. embedding the nodes of a graph or entire (sub-)graphs (Hamilton et al. 2018). Therefore, in this section we present a representation learning algorithm that addresses three aspects. First, it enables to encode graphs that represent fab states with embeddings of lower dimensionality and thus allows to handle the computational complexity of graphs. Second, graphs generated using the graph representation model presented in Section 3 whose size varies can be mapped to an embedding space of fixed dimensionality. Third, learning the graph embedding can be achieved offline, that is, before it is used as input for scheduling or performance analysis tools. This allows efficient exploitation of the results after the model has been trained.

In this work, we propose to use an approach called *graph2vec*, which was presented by Narayanan et al. (2017). We analyze its applicability and suitability to embed graphs representing fab states into a space of lower dimensionality. As a use case, we use the Mini-Fab presented by Kempf ().

4.1 Modeling the Mini-Fab as a Graph

The Mini-Fab presented by Kempf () is a small use case of a semiconductor manufacturing fab that contains some of their special characteristics like batch processing and re-entrant process flows, though. In the Mini-Fab, one product type is processed during six operations, where each operation requires a distinct time to be performed. Each operation is associated with one of the three process steps diffusion, ion implantation and lithography, where each process step is performed twice in total. For diffusion and ion implantation, there are two machines, while for lithography there is one machine only. Kempf () specifies aspects like breakdowns and maintenance as well. In this work, however, we focus on the manufacturing process in terms of operations that have to be performed on lots by machines only. The approach to represent fab states as graphs as presented in Section 3 can be used to model the Mini-Fab and in particular fab states of the Mini-Fab. In the corresponding graphs, there is a node for each equipment group, for each machine, for each route (in the Mini-Fab there is one route only), for each operation, for each processing time and for each lot that is currently in the Mini-Fab. These nodes are connected as specified in Section 3. In particular, some edges between nodes change when the lots move through the Mini-Fab from one operation and the corresponding machine to the next operation and the corresponding machine, thereby resulting in a sequence of subsequent graphs, i.e. fab states.

4.2 Generating Low-Dimensional Fab State Representations using *graph2vec*

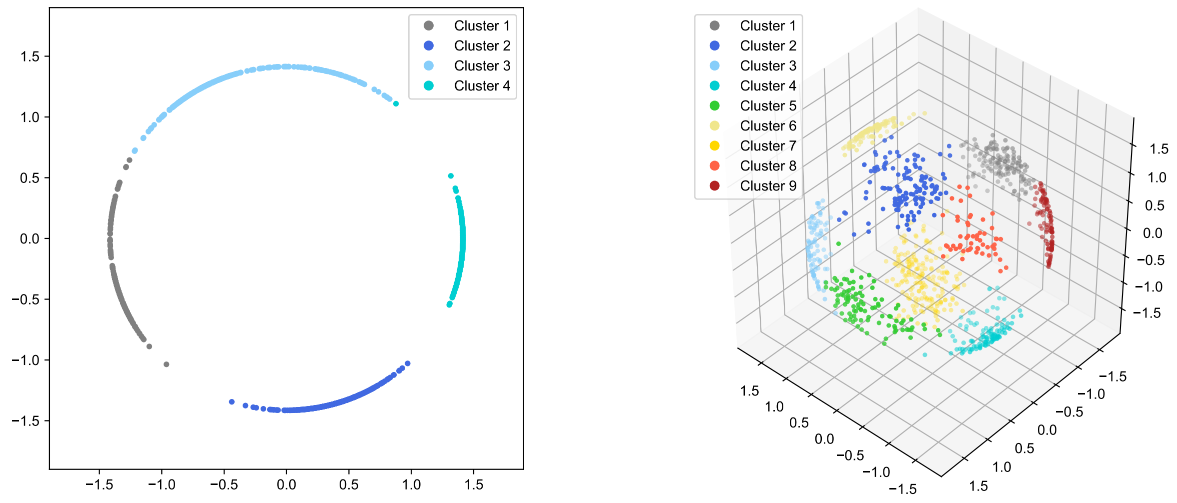
With *graph2vec*, Narayanan et al. (2017) present a neural network based approach to learn representations of graphs that is generic, unsupervised and data-driven. It is inspired by models to characterize documents based on the words and word sequences they are composed of. *graph2vec* transfers the setting of documents and words to the setting of graphs and subgraphs by characterizing graphs based on the subgraphs they are composed of. Therefore, *graph2vec* consists of two components: A procedure to extract subgraphs from a graph and a procedure to learn embeddings of different graphs based on their subgraphs. To extract the subgraphs of a graph, the neighborhood of each node in the graph is examined. This neighborhood is a set of nodes that are connected to the base node. This way of differentiating graphs based on their subgraphs allows for an aggregated perspective instead of focusing on all single nodes of a graph and thus to handle extensive graphs as well. In particular, a mapping to an embedding space with fixed dimensionality is learned.

4.3 Computational Results

To assess the applicability and suitability of *graph2vec* to map graphs representing fab states to embeddings in a space of lower dimensionality, we used the Mini-Fab of Kempf () and the graph representation model described in Section 3. To train the neural network of *graph2vec*, training data in terms of different graphs representing fab states is required. Therefore, we generated a set of 1,000 graphs (i.e. fab states) that differ in the distribution of lots in the Mini-Fab. The training was performed for both a two- and a three-dimensional embedding space. Each data point in Figure 3a and Figure 3b corresponds to one graph and thus represents one fab state. In the two-dimensional embedding space, all data points are located on a circle and four clusters can be differentiated. With regard to the meaningfulness of the embeddings and these clusters, we analyzed the fab states that are associated with the data points in the four clusters and identified a clear pattern:

- Cluster 1 represents fab states where all lots are in different operations on different machines.
- Cluster 2 contains fab states where more than one lot is in the same operation on any but the same machine.
- Cluster 3 represents fab states where more than one lot is processed on any machine but different operations are performed.
- Cluster 4 contains fab states where more than one lot is in the same operation but they are performed on different machines.

Thus, the two-dimensional embeddings of fab states of the Mini-Fab allow to differentiate between fab states of different nature although they are less complex than extensive specifications of these fab states. When a three-dimensional embedding space is used, all data points are located on a sphere and there are nine clusters of data points. Analyzing the fab states in the clusters shows that the clusters from the



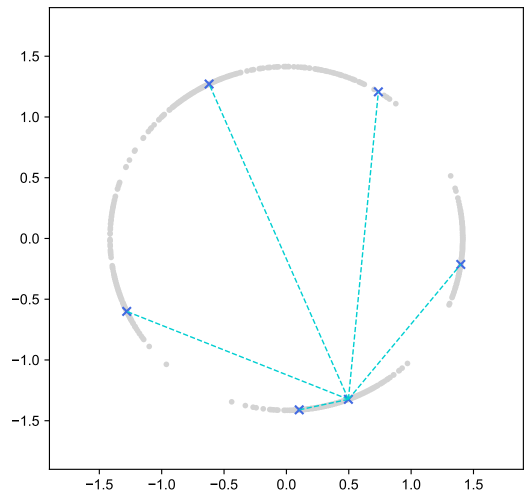
(a) 2-dimensional space (b) 3-dimensional space

Figure 3: Clustered embeddings in low-dimensional space.

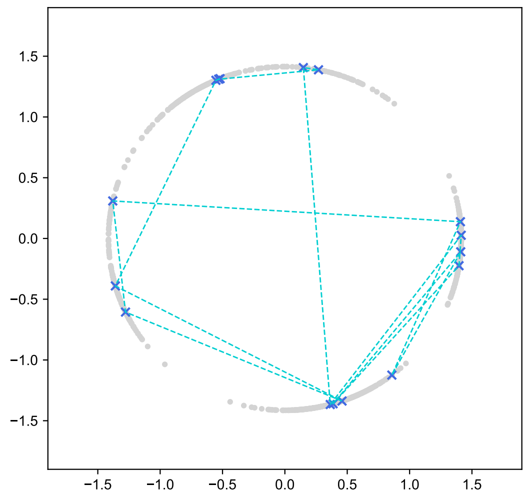
two-dimensional embedding space are split up in the three-dimensional embedding space. Thus, increasing the dimensionality of the embedding space to three dimensions allows for a more specific differentiation of fab states.

While in Figure 3a and Figure 3b the embeddings of many possible fab states are shown, another perspective is to analyze neighboring fab states, i.e. fab states that transition from one to another based on the flow of the lots through the fab. For the two-dimensional feature embedding space, Figure 4a shows the embeddings of one exemplary fab state and all its possible subsequent fab states when one lots transitions to its next operation. The wide spread of the embeddings of possible subsequent fab states indicates that – for this use case with the Mini-Fab – the embeddings of neighboring fab states are not necessarily close to each other and the structure of the underlying graph might change considerably. This is also illustrated by an exemplary sequence of embeddings representing successive fab states when a set of lots moves through the fab from the first to the last operation (Figure 4b). During processing the set of lots, the structure of the graph representing the fab state changes considerably several times. This is an important finding as it indicates that consecutive fab states which are clearly distinguishable can be mapped to embeddings that are far away from each other. This effect is also affected by the small size of the Mini-Fab model, where one lot movement yields great structural changes in the graph representation. In more complex models, we expect consecutive fab state embeddings to be closer to each other.

In total, these results demonstrate that the state of a semiconductor manufacturing fab (in this case the Mini-Fab) can be embedded into a low-dimensional space preserving some structural information of the original fab state (encoded in the features of the embedding). On the one hand, this encourages to incorporate additional features and characteristics of semiconductor manufacturing fabs into the graph representation model presented in this work. On the other hand, this motivates to apply this approach to larger semiconductor manufacturing fabs.



(a) Subsequent fab states for exemplary fab state



(b) Exemplary flow of lots through the fab

Figure 4: Embeddings representing consecutive fab states.

5 CONCLUDING REMARKS

In this work, we present a graph representation model for semiconductor manufacturing fabs that is based on the generic data model presented by Laipple et al. (2018). The graph representation model captures the interdependencies of machines, lots, routes and several more entities in the semiconductor manufacturing process, instead of extracting a set of independent features. We introduce graph representations for essential procedures in semiconductor manufacturing processes, namely process flows, material transfer, setup and maintenance activities. Finally, we use representation learning to embed graph snapshots representing fab states into a low-dimensional space. We find the low-dimensional embeddings to allow a meaningful interpretation of the underlying fab states. In view of other use cases that have been addressed successfully with a related approach (e.g. Ahn and Park (2021)), this indicates the potential to use the embeddings as input for a scheduling engine or a performance evaluation tool.

The proposed graph representation model is still subject to some limitations. For example, additional information that is important for performance analyses and production steering (e.g. due dates of lots) should be integrated in the graph representation model. Moreover, the approach presented in this work should be applied to larger semiconductor manufacturing fabs. Both aspects are part of the current research focus of the authors.

ACKNOWLEDGEMENTS

This work has received funding from the Eureka ITEA-4 Cluster AI Call 2020 under grant agreement no. 20212 (project AISSI – Autonomous Integrated Scheduling for Semiconductor Industry), which is financially supported by the Federal Ministry for Economic Affairs and Climate Action in Germany and Enterprise Singapore. This support is gratefully acknowledged.

REFERENCES

Ahn, K., and J. Park. 2021. "Cooperative Zone-Based Rebalancing of Idle Overhead Hoist Transportations Using Multi-Agent Reinforcement Learning with Graph Representation Learning". *IISE Transactions* 53(10).

Bhagat, S., G. Cormode, and S. Muthukrishnan. 2011. *Node Classification in Social Networks*. Boston, MA: Springer US.

Chen, X., S. Jia, and Y. Xiang. 2020. "A Review: Knowledge Reasoning over Knowledge Graph". *Expert Systems with Applications* 141.

Chen, Z., Y. Wang, B. Zhao, J. Cheng, X. Zhao, and Z. Duan. 2020. "Knowledge Graph Completion: A Review". *IEEE Access* 8.

Chien, C.-F., S. Dauzère-Pérès, J. W. Fowler, Z. Jiang, S. Krishnaswamy, T.-E. Lee, L. Mönch, and R. Uzsoy. 2011. "Modelling and Analysis of Semiconductor Manufacturing in a Shrinking World: Challenges and Successes". *European Journal of Industrial Engineering* 5(3).

D-SIMLAB 2022. "D-SIMCON Forecaster". http://www.d-simlab.com/category/d-simcon/products-d-simcon/forecaster-and-scenario-manager/, accessed 2022/07/06.

De, S., and A. Lee. 1998. "Towards a Knowledge-Based Scheduling System for Semiconductor Testing". *International Journal of Production Research* 36(4).

Drießel, R., and L. Mönch. 2012. "An Integrated Scheduling and Material-Handling Approach for Complex Job Shops: A Computational Study". *International Journal of Production Research* 50(20).

Geng, N., and Z. Jiang. 2009. "A Review on Strategic Capacity Planning for the Semiconductor Manufacturing Industry". *International Journal of Production Research* 47(13).

Gupta, A. K., and A. I. Sivakumar. 2006. "Job Shop Scheduling Techniques in Semiconductor Manufacturing". *The International Journal of Advanced Manufacturing Technology* 27.

Hamilton, W. L., R. Ying, and J. Leskovec. 2018. "Representation Learning on Graphs: Methods and Applications". *arXiv:1709.05584*.

Hoff, P. D., A. E. Raftery, and M. S. Handcock. 2002. "Latent Space Approaches to Social Network Analysis". *Journal of the American Statistical Association* 97(460).

Kempf, K. "Intel Five-Machine Six Step Mini-Fab Description". http://aar.faculty.asu.edu/research/intel/papers/fabspec.html.

Kim, H., D.-E. Lim, and S. Lee. 2020. "Deep Learning-Based Dynamic Scheduling for Semiconductor Manufacturing with High Uncertainty of Automated Material Handling System Capability". *IEEE Transactions on Semiconductor Manufacturing* 33(1).

Laipple, G., S. Dauzère-Pérès, T. Ponsignon, and P. Vialletelle. 2018. "Generic Data Model for Semiconductor Manufacturing Supply Chains". In *Proceedings of the 2018 Winter Simulation Conference*, edited by M. Rabe et al.

Lee, K.-H., and J. Favrel. 1985. "Hierarchical Reduction Method for Analysis and Decomposition of Petri Nets". *IEEE Transactions on Systems, Man, and Cybernetics* SMC-15(2).

Li, L., Z. Sun, M. Zhou, and F. Qiao. 2013. "Adaptive Dispatching Rule for Semiconductor Wafer Fabrication Facility". *IEEE Transactions on Automation Science and Engineering* 10(2).

Mahdavi, S., S. Khoshraftar, and A. An. 2020. "Dynamic Joint Variational Graph Autoencoders". In *Machine Learning and Knowledge Discovery in Databases*, edited by P. Cellier and K. Driessens.

Mosinski, M., T. Weissgaerber, S. L. Low, B. P. Gan, and P. Preuss. 2017. "An Easy Approach to Extending a Short Term Simulation Model for Long Term Forecast in Semiconductor Industry". In *Proceedings of the 2017 Winter Simulation Conference*, edited by W. K. V. Chan et al.

Mönch, L., J. W. Fowler, and S. J. Mason. 2013. *Production Planning and Control for Semiconductor Wafer Fabrication Facilities - Modeling, Analysis, and Systems*. New York: Springer.

Narayanan, A., M. Chandramohan, R. Venkatesan, L. Chen, and Y. Liu. 2017. "Graph2vec: Learning Distributed Representations of Graphs". *arXiv:1707.05005*.

Nickel, M., K. Murphy, V. Tresp, and E. Gabrilovich. 2016. "A Review of Relational Machine Learning for Knowledge Graphs". *Proceedings of the IEEE* 104(1).

Nyhuis, F., and P. Alves. 2002. "Methods and Tools for Dynamic Capacity Planning and Control". *Gestão & Produção* 9(3):245–260.

Ovacik, I. M., and R. Uzsoy. 1997. *Decomposition Methods for Complex Factory Scheduling Problems*. New York: Springer.

Qiao, F., Y. Ma, and X. Gu. 2013. "Attribute Selection Algorithm of Data-Based Scheduling Strategy for Semiconductor Manufacturing". In *IEEE International Conference on Automation Science and Engineering*.

Sarin, S. C., A. Varadarajan, and L. Wang. 2011. "A Survey of Dispatching Rules for Operational Control in Wafer Fabrication". *Production Planning & Control* 22(1).

Schelthoff, K., C. Jacobi, E. Schlosser, D. Plohmann, M. Janus, and K. Furmans. 2022. "Feature Selection for Waiting Time Predictions in Semiconductor Wafer Fabs". *IEEE Transactions on Semiconductor Manufacturing* 35(3).

Scholl, W., M. Förster, P. Preuss, A. Naumann, B. P. Gan, and P. Lendermann. 2016. "Simulation-Enabled Development Lot Journey Smoothening in a Fully Utilized Semiconductor Manufacturing Line". In *Proceedings of the 2016 Winter Simulation Conference*, edited by T. Roeder et al.

Scholl, W., M. Mosinski, B. P. Gan, P. Lendermann, D. Noack, and P. Preuss. 2012. "A Multi-Stage Discrete Event Simulation Approach for Scheduling of Maintenance Activities in a Semiconductor Manufacturing Line". In *Proceedings of the 2012 Winter Simulation Conference*, edited by C. Laroque et al.

Scholl, W., D. Noack, O. Rose, B. P. Gan, P. Lendermann, P. Preuss, and F. S. Pappert. 2011. "Implementation of a Simulation-Based Short-Term Lot Arrival Forecast in a Mature 200mm Semiconductor Fab". In *Proceedings of the 2011 Winter Simulation Conference*, edited by S. Jan et al.

Seidel, G., C. F. Lee, A. M. Kam, B. P. Gan, C. W. Chan, A. Naumann, and P. Preuss. 2017. "Harmonizing Operations Management of Key Stakeholders in Wafer Fab Using Discrete Event Simulation". In *Proceedings of the 2017 Winter Simulation Conference*, edited by W. K. V. Chan et al.

Seidel, G., C. F. Lee, A. Y. Tang, S. L. Low, B. P. Gan, and W. Scholl. 2020. "Challenges Associated with Realization of Lot Level Fab Out Forecast in a Giga Wafer Fabrication Plant". In *Proceedings of the 2020 Winter Simulation Conference*, edited by K.-H. Bae et al.

Seidel, G., P. Preuss, C. Canbolat, S. L. Low, C. W. Chan, and B. P. Gan. 2019. "An Integration of Static and Dynamic Capacity Planning for a Ramping Fab". In *Proceedings of the 2019 Winter Simulation Conference*, edited by N. Mustafee et al.

Zhang, M. T., J. Fu, and E. Zhu. 2008. "Dynamic Capacity Modeling in Semiconductor Assembly Manufacturing". *International Journal of Production Research* 46(3).

Zhou, M., and M. D. Jeng. 1998. "Modeling, Analysis, Simulation, Scheduling, and Control of Semiconductor Manufacturing Systems: A Petri Net Approach". *IEEE Transactions on Semiconductor Manufacturing* 11(3).

Zitnik, M., and J. Leskovec. 2017. "Predicting Multicellular Function Through Multi-Layer Tissue Networks". *Bioinformatics* 33(14).

AUTHOR BIOGRAPHIES

BENEDIKT SCHULZ* is research associate at the Institute for Material Handling and Logistics (IFL) at Karlsruhe Institute of Technology (KIT) in Karlsruhe, Germany. His research interests are stochastic models to design and analyze production and logistic systems as well as data science. His email address is benedikt.schulz@kit.edu.

CHRISTOPH JACOBI* is research associate at the Institute for Material Handling and Logistics (IFL) of the Karlsruhe Institute of Technology (KIT) in Karlsruhe, Germany. His research interests are in stochastic modeling and performance analysis as well as scheduling of stochastic systems. His e-mail address is christoph.jacobi@kit.edu.

ANDREJ GISBRECHT is a simulation engineer at the Robert Bosch GmbH. He received his Diploma from the Clausthal University of Technology and his PhD in computer science from the Bielefeld University. He currently works on artificial intelligence and numerical optimization techniques in the area of semiconductor fabrication. His email address is andrej.gisbrecht@de.bosch.com.

EVANGELOS ANGELIDIS is a simulation engineer and IT-architect of the semiconductor operations team (SCO) by Robert Bosch GmbH. He received his M.S. degree in computer science from Dresden University of Technology. He has years of experience in developing simulation and optimization software in the domain of production and logistics. His research focuses on the simulation and optimization of complex assembly lines such as semiconductor and aerospace manufacturing. His email address is evangelos.angelidis@de.bosch.com.

CHEW WYE CHAN is a Software Engineer of D-SIMLAB Technologies (Singapore). He holds a Master of Computing degree from the National University of Singapore. He is currently working as a doctoral student at the School of Computer Engineering at Nanyang Technological University, Singapore. His research interests include machine learning, data science, and simulation-based optimization. His email address is chew.wye@d-simlab.com.

BOON PING GAN is the CEO of D-SIMLAB Technologies (Singapore). He has been involved in simulation technology application and development since 1995, with primary focus on developing parallel and distributed simulation technology for complex systems such as semiconductor manufacturing and aviation spare inventory management. He led a team of researchers and developers in building a suite of products in solving wafer fabrication operational problems. He was also responsible for several operations improvement projects with wafer fabrication clients which concluded with multi-million dollar savings. He holds a Master of Applied Science degree, specializing in Computer Engineering. His email address is boonping@d-simlab.com.

*These authors contributed equally to this work.