Task 1: Top Conferences and Journals

Conferences

Name	**Type**	**Relevance**	**CORE** [1]	**ERA** [2]	**Acceptance Rate (past 5 years)** [3]
International Joint Conference on Artificial Intelligence	Conference	[4] is a seminal paper that was originally presented at this conference. It presents a new data structure for the representation of Boolean functions – the Sentential Decision Diagram (SDD). This data structure is a generalisation of the Ordinary Binary Decision Diagram (OBDD) and has many use cases in knowledge representation and symbolic model checking, as outlined in this paper. Therefore, this paper provides a motivation for one of our research problems, which is to determine whether there exists an exponential increase in the complexity of a problem when its input is represented as an SDD. This result has been shown for other symbolic representations such as Boolean circuits, formulae and OBDDs [5], [6], [7].	A*	A	15%
International Colloquium on Automata, Languages, and Programming	Conference	[5] is a core paper for our research as it proves the general result that problems which are hard for a particular complexity class under a log time (LT) reduction become hard for the complexity class that is exponentially more difficult when the input is represented with a Boolean circuit. This is extremely important for our project as we are trying to adapt this proof, to show a similar result for when the input is represented with an SDD. This core paper was originally presented at this conference.	A	A	29%

IEEE Conference on Computational Complexity	Conference	[7] is also a core paper as it established a similar result to [5] – but for problems where the input is represented with an OBDD instead. More specifically, the paper finds that the original problem must be hard under "quantifier free" reductions (as opposed to LT reductions) in order for the succinct problem to be exponentially more difficult. It is noted in this paper that the required "quantifier free" reductions are "weaker" than "log time" reductions in the sense that all quantifier free reductions are also log time reductions. It is postulated that this is because OBDDs are less succinct than circuits. Thus, the paper gives us insight into the strength of the reduction that is required for to attain the same result for SDDs. This core paper was originally presented at this conference.	A	B	35%
International Symposium on Mathematical Foundations of Computer Science	Conference	The auxiliary paper [8] was originally presented at this conference. [8] is important for our research, as it adapts the proofs shown in [5] and [7] to show a similar general exponential increase in problem complexity when the inputs are represented with a logic on "bit-vectors". This is important for our project as we can examine how the seminal proofs were adapted for a bit-vector representation, so that we have a better idea of how these proofs can be adapted for a SDD representation.	A	A	34%
Current Trends in Theory and Practice of Computer Science	Conference	This journal is relevant as paper [9] was published here. This paper is important, as it proves complexity results for *particular* graph problems when the input is represented as an OBDD. In particular, the Eulerian Cycle, Bipartiteness and Planarity problems are proven to be PSPACE-complete when the input is represented as an OBDD. One of our research questions is to determine the complexity of the alternating finite automata (AFA) emptiness problem when the input is represented as a Boolean formula/circuit. Thus, this paper gives us ideas on how one can adapt the complexity proofs from *particular* explicit problems to determine the complexity of their equivalent symbolic problems.	B	B	33%

Information Processing Letters	Journal	[6] is a core paper that was published in this journal. The paper is important for our research as it established a similar result to [5] – but for problems where the input is represented with a Boolean formula instead. The paper also finds that the original problem must be hard under “quantifier free” reductions (as opposed to LT reductions) in order for the equivalent succinct problem to be exponentially more difficult. Again, this is similar to our task of proving a similar result for problems where the input is represented by an SDD.	B	B	-
Information and Computation	Journal	[10] and [11] are both seminal papers for this project, as they are the first two papers to discuss the complexity of problems where Boolean circuits are used to represent the input. [10] shows that a particular class of graph problems which are normally L-hard, become NP-hard when the graph is represented by a Boolean circuit. [11] then shows that if SAT reduces to a problem under a “projection” reduction – then the succinct problem becomes NEXP-hard. Even though these results were then superseded by the result in [5], these papers were still the first to investigate the use of Boolean circuits for the representation of graphs. [12] is also an auxiliary paper published to this journal. It explores how the complexity of problems changes when the input is expressed with Conjunctive Normal Form (CNF) or Disjunctive Normal Form (DNF) Boolean formulae. This is significant as it is one of the first papers to show that a exponential increase in the difficulty of a problem *doesn’t* always occur when the input is represented symbolically. In other words, it was shown that many problems remain at the same complexity when their input is expressed with a CNF or DNF formula. This is important as it shows that for some types of symbolic representation, a general exponentiation of the complexity class doesn’t exist.	B	B	-

Artificial Intelligence	Journal	[13] is an important auxiliary paper published to this journal. It adapts the proof from [5] for a slightly different model of symbolic representation. The model is still a circuit, however, this circuit represents the adjacency list of the graph as opposed to the adjacency matrix – as it did in [5]. This is important as it gives us ideas of how we can change the proof from [5] to fit our own model of symbolic representation – SDDs. [14] was also published in this journal. This paper determines the complexity of propositional planning problems with different constraints. Firstly, this provides a motivation for us to study using Boolean formulae as a method of succinct representation – as propositional planning is extremely similar. Secondly, it also provides us with complexity results on planning problems that we can possibly adapt to obtain complexity results on succinct Boolean formula automata problems.	A*	A*	-
Acta Informatica	Journal	[15] explores non-Boolean succinct representations of graphs. It explores “Integer Expressions” and “General Hierarchic Input Language” as methods of succinct representation. These methods are used in VLSI (Very Large-Scale Integration) to represents sets succinctly by defining them as the combination of pre-defined sets. It is found that particular problems (such as the graph accessibility problem) experience an exponential increase in complexity when the input is represented using these methods. This paper demonstrates that similar results such as the one from [5] may apply to non-Boolean symbolic representations of graphs. This may be useful if we decide to extend our study to non-Boolean succinct representations of graphs.	B	A	-
Journal of Computer and System Sciences	Journal	[16] explores the use of hierarchical graph models (which are often used in CAD) to succinctly represent graphs. Similar to [12], it is found that the complexity of many graph problems does not change when the input is represented hierarchically – however certain problems do indeed exhibit this exponential increase in difficulty. This is important as it shows that for some types of symbolic representation, a general exponentiation of the complexity class doesn’t exist when inputs are given symbolically.	A*	A*	-

Chicago Journal of Theoretical Computer Science	Journal	[17] is a seminal paper that was published into this journal. The paper is the first to show a general exponential increase in the complexity of problems when their input is represented with OBDDs. This is significant as it was the first paper to prove a result similar to [5] on a method of succinct representation that wasn't circuits. This is relevant to us as it gives us more ideas of how we can generalise this result to problems which take *other* symbolic representations (such as SDDs) as input.	C	C	-
Theory of Computing Systems	Journal	In this journal, paper [18] explores a similar idea to paper [12] – showing that a general exponentiation in the complexity of a problem doesn't occur when the input is represented with a Boolean circuit with depth 2, since many problems remain at the same complexity when the input is expressed in this manner. However, for circuits with depth 3 or higher, such a generalisation can indeed be found. [19] was also published to this journal, exploring the succinctness of SDDs in comparison to other data structures such as OBDDs and circuits. This is important for our project as it allows us to better understand which aspects of the existing proofs for circuits and OBDDs in [5], [7] work/don't work for SDDs.	C	C	-

Note: Even though C ranked journals are provided, this is because seminal papers were published in them, making them highly relevant for our project.

Task 2: Top Research Groups

Name	General Information	Relevance	Location	Number of People
Laboratory for Relational Algorithmics, Complexity and Learning (LARCA)	LARCA is a research group associated with the tertiary education institution - Barcelona Tech. As such, most of its members are professors at the university and the post-graduate students they supervise. They focus their research on machine learning and large data analytics, and also have a strong focus on using mathematical tools from complexity theory, logic and automata theory to approach problems. While the group does receive funding from companies and institutions to conduct research, their primary objective is to foster collaboration between group members.	Balcazar was the (co) author of the seminal papers [5], [13], [20] all of which detail a general exponentiation in the complexity of problems when their inputs are represented with Boolean circuits. Currently, Balcazar is the coordinator of the research group LACRA. Even though his seminal papers were published before his joining of the research group, he continues to actively publish in collaboration with other members of the group on topics regarding algorithmic learning theory.	Barcelona	13 full-time researchers and ~30 under/postgraduate students

Logic and Programming (LOGPROG)	LOGPROG is another research group which is associated with Barcelona Tech. As such, most of its members are postgraduate students or teaching staff from the Languages and Informatics department at the university. They specialise on the applications of Logic in computer science, such as Satisfiability Modulo Theory (SMT), automated theorem proving and descriptive complexity. The group often participates in research projects which are funded by the EU/Spanish Government, and also organises seminars and workshops on Logic for other members of the university. Their primary objective is collaboration and spreading interest in the uses of logic in computer science.	Lozano was the (co) author of the same aforementioned papers [5], [20] with Balcazar as his PhD supervisor. Lozano is currently a member of LOGPROG. Even though his seminal papers were published before he joined LOGPROG, he actively publishes with other members of the research group on knowledge representation with logic.	Barcelona	7 full-time researchers, and their postgraduate students
The OVERLAY (Formal Verification, Logic, Automata and Synthesis) Research Group for Formal Methods in AI	OVERLAY is a collection of full-time researchers that specialise in the use of Formal Methods in Artificial Intelligence. Their goal is to foster collaboration and knowledge-sharing between the Italian researchers in this field. They do this by organising an annual workshop which highlights the developments at the intersection of these two fields.	Faella co-authored the paper [21] which outlines how to solve reachability games which are represented succinctly with first order logic. The applications of solving these games outlined in the paper therefore provides us with a motivation to study succinct representations of alternating automata (since they have similar semantics to reachability games). Faella is currently a member of OVERLAY, where he participates in workshops organised by the research group and frequently publishes with other members of the group.	Italy	40 full-time researchers

The Modelling and Verification Research Group at IRIF (Fundamental Computing Research Institute) Laboratory	The Modelling and Verification Research Group is a collection of professors and the PhD students they supervise at IRIF – a French research institute associated with the University of Paris City. They focus on system verification/model checking, and the different algorithms used to achieve this. Again, their goal is to foster collaboration amongst group members – and to achieve this, weekly seminars are given by members.	Parlato co-authored the paper [21] with Faella. He is currently a professor at IRIF, and is a permanent member of the Modelling and Verification Research Group, where he actively publishes papers with other members of the group and presents seminars to other members on his findings.	Paris	14 full-time researchers and 16 PhD students
FORSYTE Group at Vienna University of Technology	The FORSYTE Group is a collection of professors and the PhD students they supervise at Vienna University. They specialise in model checking, synthesis, and verification. Outside of offering courses in this field to the University, the research group aims to foster collaboration between researchers and supervise enthusiastic Masters/PhD students. They also regularly host talks and seminars at the University to promote interest in logic and algorithms amongst students.	Veith authored two seminal papers. [6] analysed the exponential increase in difficulty of problems when input is represented with a Boolean formula, while [7] proved a similar result for OBDDs. Up until near his passing, Veith was a member of FORSYTE, taught at Vienna university, and published a total of 14 papers with the group.	Vienna	24 full-time researchers and 4 PhD students

The Learning and Intelligent Systems Group at MIT	The LIS Group is a group of post-graduate MIT research students lead by two principal investigators: Kaelbling and Lozano. The research group focuses on using AI planning, machine learning and reinforcement learning to design intelligent robot systems. The principal researchers also work on promoting interesting developments in the field of intelligent robotic systems, holding seminars and writing blog posts on findings from the group.	Kaelbling authored an auxiliary paper which proved the complexity of hierarchical planning in AI [22]. We previously adapted this proof, allowing us to show that a similar problem regarding succinct Boolean NFAs had the same complexity. Kaelbling is one of two principal researchers at the LIS group, where she supervises a group of PhD students. Over her career, she has published over 100 papers with students from this group.	Massachusetts	2 full-time researchers and 30+ postgraduate students

Task 3: Exemplary Papers

Paper 1: The Complexity of Algorithmic Problems on Succinct Instances [5]

Brief overview of paper:

We firstly define the following terminology:

- We say that a problem A is hard for a complexity class C under a class of reductions R if every problem in C reduces to A using a reduction from R.
- For a computational problem A, its equivalent succinct circuit problem sA is the set of all Boolean circuits that represent true instances of A.

Balcazar et al show that if a problem A is hard for complexity class C under log time (LT) reductions, then sA is hard for the complexity class EXPC under polynomial time reductions. In this case, a log time reduction f is a reduction which has the property that a log-time bounded indexing Turing machine can compute $f(x)_i$ when given x and i as input.

Justification of excellence:

The paper is exemplary for many reasons. Firstly, the main result achieved in this paper is much more general than previous, similar results.

- For instance, Galperin et al [10] showed that graph properties which are "t-critical" become NP-hard to determine when the graph is represented with a Boolean circuit. However, all the NP-hardness results from [10] can also be achieved using the result from this paper, showing that Balcazar et al prove a more general result.
- Another example is that Papadimitriou et al [11] show that if there is a projection reduction from SAT to problem A, then sA is NEXP-hard. Balcazar's result is more general because projection reductions are a strict subset of LT reductions, meaning that this exponential increase in complexity is now proven for a greater class of problems. Furthermore, Balcazar's result generalises for *any* complexity class that A is hard for, while Papadimitriou's proof must be tediously adapted to prove a similar result for each complexity class.

Thus, it is clear that this paper makes significant improvements to the previous results in the field. Furthermore, the paper is especially beneficial for our research project, as we wish to prove that a similar exponential increase in the hardness a problem occurs when the input is represented by a Sentential Decision Diagram (SDD). Since SDDs are very similar to Boolean circuits, the paper shows us that the result we wish to prove is indeed plausible. Furthermore, it gives us a scaffold of a proof that we can possibly try to adapt for our particular symbolic representation. Finally, the writing style and structure of this paper is excellent, as the previous work on succinct representations is very clearly outlined – and the practical applications and motivations for this work is also clearly defined at the beginning. Furthermore, Balcazar et al provides intuitions and examples before providing exact proofs and definitions – allowing for easier comprehension by the reader.

Paper 2: How to encode a logical structure by an OBDD [7]

Brief overview of paper:

We firstly define the following terminology:

- For a computational problem A, its equivalent succinct OBDD problem sA is the set of all OBDDs that represent true instances of A.

Veith shows a similar result to Balcazar et al, except that the method of symbolic representation is now OBDDs instead of circuits. More precisely, Veith shows that if a problem A is hard for complexity class C under quantifier free reductions, then the sA is hard for the complexity class $EXPC$ under polynomial time reductions. In this case, a reduction I (in descriptive complexity) between two problems $T \subseteq Struct(\tau),\ \ S \subseteq Struct(\sigma)$ is a set of first order sentences over the signature τ, such that $A \in T \Leftrightarrow I(A) \in S$. If these first order sentences have no quantifiers, then the reduction is quantifier free.

Justification of excellence:

Again, the main result achieved in this paper improves upon previous results on the complexity of succinct OBDD problems. Feigenbaum et al [17] showed that if there is a NC0 reduction from SAT to problem A, then sA is NEXP-hard. Veith's result strictly improves upon this because quantifier free reductions are a strict superset of NC0 reductions. Furthermore, Veith's result generalises for *any* complexity class that problem A is initially hard for, while Feigenbaum's proof must be tediously adapted to prove a similar result for each complexity class.

In addition to this, this paper gives us further insight into our research problem. Firstly, it further reinforces that our first research problem is indeed feasible. This is because SDDs are a generalisation of OBDDs, meaning that it is certain that we can prove a similar result for SDDs if we directly translate the proof from this paper. However, this paper is also beneficial for our project as it reveals a general trend between the succinctness of a particular symbolic representation and the "strength" of the result that can be proved. In particular, it is observed that OBDDs are less succinct than Boolean circuits, in the sense that an OBDD can be converted to a circuit with a polynomial increase in size, however, certain circuits require an exponentially larger OBDD to represent the same function. Therefore, the reduction that is required in the result is more restricted for OBDDs (quantifier free) in comparison to circuits (LT), in the sense that all quantifier free reductions are also LT reductions. Therefore, this allows us to postulate that since SDDs are not as succinct as circuits, but more succinct than OBDDs, a class of reductions which is a subset of LT reductions, but a superset of quantifier free reductions should be used in our proof. Finally, this paper is also exemplary as Veith precisely and succinctly states all the required definitions before providing proofs and results, meaning that all the required knowledge is provided in the paper, but no any unnecessary information is provided.

Task 4: Unanswered Research Problems

Research Problem 1:

Problems on graphs traditionally assume that the size of input is polynomial with respect to the size of the graph, as is the case with adjacency lists and matrices. However, the graphs that appear in fields such as model checking [23], [24], [25] and AI planning [14] tend to have a very large number of vertices. As such, data structures like Boolean formulae, circuits and OBDDs (Ordered Binary Decision Diagrams) [26] are often used to encode these graphs with space that is - in the best case logarithmic, and in the worst case linear in size with respect to the size of the graph.

We say that a graph $G = (V, E)$ is succinctly represented by a Boolean structure (formula, circuit, OBDD), if, when two binary integers i, j are given as input to the Boolean structure, the corresponding (i, j) entry of the adjacency matrix for G is outputted. Thus, for each computational problem A, we can define the succinct formula/circuit/OBDD version of A (sA) to be the set of formulae/circuits/OBDDs which describe true instances of A. Previous works on Boolean circuits [5], [11] formulae [6] and OBDDs [7], [17] have all found the generalisation that if a problem A is hard for complexity class C under a particular reduction, then the problem sA is hard for the complexity class one exponent above C. In the case of circuits, the required reduction is a polylogarithmic time reduction, while in the case of formulae and OBDDs, the required reduction is a quantifier free reduction.

Recently, SDDs (Sentential Decision Diagrams) – a generalisation of OBDDs, have been defined to represent Boolean functions more succinctly than OBDDs, in the formal sense that there are Boolean functions that can be represented with a polynomial size SDD, but require an exponential size OBDD [4], [19]. This has led to their increased use in knowledge representation and symbolic model checking [27]. However, despite their increased usage, it is still unknown whether succinct SDD problems experience the same general exponential increase in complexity that was previously found for other methods of symbolic representation. Our goal is to prove that a similar result holds for succinct SDD problems. It is certain that a problem which is hard under quantifier free reductions will have a corresponding succinct SDD problem which is exponentially harder - as SDDs are a generalisation of OBDDs, meaning that we can obtain the result from [7]. However, we postulate that having a problem be hard under a more *general* reduction (of which, quantifier free reductions are a subset) will suffice for the corresponding succinct SDD problem to become exponentially harder. We believe this to be the case as SDDs are more succinct than OBDDs [19].

To achieve our goal, we will replicate the proofs from [5], [7], [6], [11], [17], modifying the strength of the reduction for which the original problem A is hard under, until we are able to prove the required result for SDDs.

Research Problem 2:

The reachability problem in the verification of (finite) open systems is as follows:

Consider a finite directed graph $G = (V, E)$ where V is partitioned into V_1 and V_2, and $B \subset V$ is a set of goal states. A token is placed at a given initial vertex and two players move the token in the following fashion. If the token is on a vertex $v_1 \in V_1$ then player 1 moves the token along an edge outgoing from v_1, and if the token is on a vertex $v_2 \in V_2$ then player 2 moves the token along an edge outgoing from v_2. The reachability question asks whether the first player can "win" by reaching a goal state [28].

There are many real-world instances of this problem with useful applications [29], [30], [31], [32]. For instance, in the field of network architecture, solving the reachability question can determine whether an agent (player 1) is able to reach their destination (goal state) regardless of how its environment (player 2) acts. However, the graphs in these applications generally have a very large number of vertices. As such, data structures like Boolean formulae, circuits and OBDDs [26] are often used to encode these graphs with space that is - in the best case logarithmic, and in the worst case linear in size with respect to the size of the graph. (See Research Problem 1 above for details on how exactly these data structures encode the graphs). It is well known that the reachability problem is P-TIME complete, and that the succinct OBDD reachability problem is EXP-TIME complete [28], [33]. However, to the best of our knowledge, the complexity of the succinct Boolean circuit/formula reachability problem has not been explicitly published in literature.

We say that an alternating finite automaton (AFA) $A = \left(\Sigma, V_\forall, V_\exists, \delta, q_0, F\right)$ is succinctly represented by a set of Boolean formula/circuits $\left\{\Phi_a \mid a \in \Sigma\right\}$, when $\delta\left(q, a\right) = p$ if and only if $\Phi_a\left(q, p\right) = 1$ (assuming that there is an arbitrary binary enumeration of the states, and q, p are written according to this enumeration). It is now clear that the succinct Boolean circuit/formulae reachability problem is equivalent to the succinct Boolean circuit/formula AFA emptiness problem, in the precise sense that there is a trivial linear time reduction between the problems. Thus, it is our goal to determine the complexity of the succinct Boolean circuit/formula AFA emptiness problem. To do this, we firstly recall that [5], [6] show a general relation between a problem A and its succinct problem sA for Boolean circuits/formulae, given that A is hard under polylogarithmic time/quantifier free reductions respectively. Therefore, to utilise these results to achieve our goal, we will try to prove that the AFA emptiness problem is indeed P-TIME hard *under these reductions*. If this fails, then we will try to adapt the original P-TIME hardness proof for AFA emptiness for the succinct problem. This is plausible, as we have constructed similar proofs on succinct Boolean formula problems on non-deterministic finite automata (NFAs) in the past.

Task 5: Annotated Bibliography

Research Problem 1

[10] H. Galperin and A. Wigderson, "Succinct representations of graphs," *Information and Control*, vol. 56, no. 3, pp. 183–198, Mar. 1983, doi: 10.1016/S0019-9958(83)80004-7.

At a high level, in the paper, Galperin et al define a way to represent graphs with Boolean circuits that is inspired by VLSI (Very Large Integrated Circuitry) design, and then find that determining whether a graph (represented as a circuit) has a simple property such as connectedness becomes NP hard.

This paper firstly defines the succinct representation of a graph $G = (V, E)$ with vertices $v_0 \dots v_n$, as a circuit C_G which takes as input two binary numbers i, j and outputs 1 iff $\left(v_i, v_j\right) \in E$. This succinct representation of a graph has size which is $O(\log\left(\left|V\right| + E\right))$ in the best case, and $O(\left|V\right| + E)$ in the worst case. The key result of this paper is that if a graph property Q is "t-critical" for all integers $t \geq 1$, then the problem $\{circuit\ C \mid C\ represents\ a\ graph \in Q\}$ is NP-hard. Here, a graph property Q being "t-critical" for some integer t, means that there exists a graph with $t + 1$ nodes (where one node w is a dummy node) such that the graph currently does not have property Q, and also does not have any edges to/from the dummy node. However, the addition of any edge to/from the dummy node causes the graph to attain property Q. Showing that a graph property is "t-critical" is quite simple, because a property being "1-critical" is sufficient for it to be "t-critical" for all integers t. This key result is then applied to multiple graph properties, thereby showing that finding the existence of a k length cycle, k length path, vertex with degree $\geq k$ etc. all become NP hard when the graph is represented with a circuit.

This seminal paper is highly important, as it is the first theoretical investigation into representing a graph with a Boolean structure. Thus, this paper is very impactful, as a large proportion of future investigations into symbolic representations of combinatorial objects use Boolean structures (formulae, circuits, OBDDs) to do so [6], [7]. Previous investigations on succinct representations of graphs used hierarchical structures instead of Boolean structures [16]. However, in comparison to future research on the symbolic representation of graphs, the result proved here is not very general. Firstly, this is because "t-critical" graph properties do not encapsulate a large set of graph properties. Secondly, this is because only an increase in difficulty to NP hardness is shown, even though it is suspected that a similar increase in difficulty applies to graph problems belonging to other complexity classes as well.

For our research we are similarly investigating the increase in complexity of graph problems when the graphs are represented symbolically with SDDs. Even though we would ideally like to show an increase in complexity for a more general class of problems than what is shown here, if we fail to do so, we can still mimic the proof used here to show a similar result for symbolic representation using SDDs.

[11] C. H. Papadimitriou and M. Yannakakis, “A note on succinct representations of graphs,” *Information and Control*, vol. 71, no. 3, pp. 181–185, Dec. 1986, doi: 10.1016/S0019-9958(86)80009-2.

At a high-level Papadimitriou et al investigate a symbolic representation of graphs identical to the one defined by Galperin et al, and improve on the results that were previously proven. It is found that the complexity of a *“more general”* class of graph problems increase by an exponential factor when the graph is represented with a circuit.

More specifically, for a graph problem A, the paper defines the corresponding succinct problem $sA = \{circuit\ C \mid C\ represents\ a\ graph\ G \in A\}$. It is then found that if a projection reduction (to be defined later) from SAT to A exists, then sA is NEXP-hard. A projection reduction f is a reduction with the particular property that the bit at the jth index of $f(x)$ can be determined using at most one bit at the ith index of x. This index i can be computed from the index j in time logarithmic with respect to x. Using this general result, Papadimitriou shows that the problems HAMILTON-CYCLE, 3-COL etc. are all NEXP-complete for graphs represented with circuits. The paper also hints that similar theorems for other complexity classes which show that sA is PSPACE-hard, EXPSPACE-hard etc. exist, however, the authors leave this as an exercise for the reader to prove themselves.

The paper greatly improves on Galperin’s findings as not only can this new theorem be used to prove all the results from Galperin’s paper, but it can also prove many more. Furthermore, this paper is seminal as it is the first to state that the *type of reduction* under which problem A is hard for is the condition which determines if sA is hard for an exponentially higher complexity class. A similar idea is carried onwards throughout all further studies [6], [7]. However, even though the result from this paper *can* be adapted for complexity classes besides NEXP, doing so is a highly involved and tedious process.

This paper is relevant to our research, as it gives us a proof that we can try to mimic to prove a similar result for SDDs.

[5] J. L. Balcázar, A. Lozano, and J. Torán, “The Complexity of Algorithmic Problems on Succinct Instances,” in Computer Science: Research and Applications, R. Baeza-Yates and U. Manber, Eds. Boston, MA: Springer US, 1992, pp. 351–377. doi: 10.1007/978-1-4615-3422-8_30.

At a high level this paper continues to investigate a symbolic representation of graphs with Boolean circuits, however, aims to further generalise the results from Papadimitriou et al.

We say that a problem A is hard for a complexity class C under a class of reductions R if every problem in C reduces to A using a reduction from R. Now, the key result of the paper is as follows:

If a problem A is hard for complexity class C under log time (LT) reductions (to be explained later), then sA is hard for the complexity class that is exponentially harder than C under polynomial time reductions. A log time reduction f has the property that a log-time bounded indexing

Turing machine (i.e. a Turing Machine which can write a binary integer i on an additional tape to access index i on the input tape instantly) can compute $f(x)_i$ given x and i on the input tape.

This improves upon Papadimitriou's results in several ways. Firstly, the required LT reduction is more general than the previously required projection reduction, in the sense that every projection reduction is also an LT reduction – but not the other way around. Furthermore, unlike before, the generalised result shows that this exponentiation of complexity class occurs regardless of the initial complexity class of the explicit problem.

Despite this, the paper fails to provide any generalisation/adaptation of this proof to other methods of succinct representation such as Boolean formulae/OBDDs. However, it is still extremely useful in regard to our first research problem as it suffices in giving us a proof we can hope to adapt for other methods of succinct representation such as SDDs in order to find a similar result.

[17] J. Feigenbaum, S. Kannan, M. Y. Vardi, and M. Viswanathan, "Complexity of problems on graphs represented as OBDDs," in STACS 98, vol. 1373, M. Morvan, C. Meinel, and D. Krob, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 1998, pp. 216–226. doi: 10.1007/BFb0028563.

The high-level purpose of this paper is to generalise the results from Papadimitriou for symbolic representations of graphs using OBDDs.

More specifically, this paper firstly defines the succinct representation of a graph $G = (V, E)$ with vertices $v_0 \dots v_n$, as an OBDD O which takes as input two binary numbers i, j and outputs 1 iff $\left(v_i, v_j\right) \in E$. Furthermore, for a graph problem A, the paper defines the corresponding succinct problem $sA = \{OBDD\ O \mid O\ represents\ a\ graph\ G \in A\}$.

Now, the key result of the paper is that if an NC0 padding reduction exists from SAT to a graph problem A, then sA is NEXP hard. An NC0 padding reduction f has the property that there is a constant integer k such that for each j, $f(x)_j$ can be determined using k bits of x. The indices of these k bits must be able to be computed by a Mealy Machine (a DFA which can write onto a tape) when given j as input.

It is also shown that the succinct OBDD AGAP (alternating graph accessibility) problem is EXPTIME-complete. This was shown separately, as the explicit AGAP problem is not PTIME-hard *under NC0 reductions*, meaning that the general theorem could not be applied to this case. To prove this side result, Feigenbaum et al adapts the original proof for the PTIME-hardness of AGAP to show that succinct OBDD AGAP is EXPTIME-hard. Therefore, this paper is imperative in extending the established results on Boolean circuit representations of graphs to other symbolic models of graphs such as OBDDs.

This paper is relevant to us, as it gives us ideas on how we can adapt the proof from Papadimitriou et al and Balcazar et al to obtain a similar result for SDD representations of graphs. Furthermore, this paper also gives us ideas on how we can approach research problem 2, by adapting the proof of hardness from the explicit problem to show the hardness of the succinct problem.

[7] H. Veith, "How to encode a logical structure by an OBDD," in Proceedings. Thirteenth Annual IEEE Conference on Computational Complexity (Formerly: Structure in Complexity Theory Conference) (Cat. No.98CB36247), Jun. 1998, pp. 122–131. doi: 10.1109/CCC.1998.694598.

This paper slightly strengthens the results from Feigenbaum et al on the complexity of succinct OBDD problems. In particular, it is shown that if a problem A is hard for complexity class C under quantifier free reductions (to be explained later), then the sA is hard for the complexity class exponentially higher than C under polynomial time reductions. More details on quantifier free reductions can be found in [34], however a high level overview is given here. A reduction I (in descriptive complexity) between two problems $T \subseteq Struct(\tau), \quad S \subseteq Struct(\sigma)$ is a set of first order sentences over signature τ, such that $A \in T \Leftrightarrow I(A) \in S$. If these first order sentences have no quantifiers, then the reduction is quantifier free.

This improves upon the results shown by Feigenbaum et al, since it generalises the result to problems which are not NP-hard but instead hard for other complexity classes. Furthermore, it relaxes the sufficient condition for sA to be exponentially harder than A, since all NC0 padding reductions are also quantifier free reductions.

This paper is critical to us as it further enforces the feasibility of proving a similar general result to the one in this paper for SDDs. In fact, since SDDs are a generalisation of OBDDs, it is certain that a problem which is hard under quantifier free reductions will have a corresponding succinct SDD problem which is exponentially more difficult (by applying the results of this paper directly).

Furthermore, this paper also shows a trend. It is observed that OBDDs are less succinct than Boolean circuits, in the sense that an OBDD can be converted to a circuit with a polynomial blow-up, however, certain circuits require an exponentially larger OBDD to represent the same function. Therefore, the reduction that is required is more restricted for OBDDs in comparison to circuits, in the sense that all quantifier free reductions are also LT reductions. Therefore, this allows us to postulate that since SDDs are not as succinct as circuits, but more succinct than OBDDs, a class of reductions which is a subset of LT reductions, but a superset of quantifier free reductions may be required for SDDs.

[13] JoséL. Balcázar, "The complexity of searching implicit graphs," Artificial Intelligence, vol. 86, no. 1, pp. 171–188, Sep. 1996, doi: 10.1016/0004-3702(96)00014-8.

In Artificial Intelligence, a graph with n nodes $v_0 \ldots v_{n-1}$, each with degree $\leq d$ is represented symbolically with a circuit which has $\log(n)$ input gates and $dlog(n)$ output gates. The circuit C has the property that when i is inputted in binary, it outputs $j_1,\ j_2,\ j_3,\ \ldots,\ j_d$ where each j_i is outputted in binary in the ith group of $\log(n)$ output gates. The output must have the property that $v_{j_1},\ v_{j_2} \ldots,\ v_{j_d}$ are precisely the nodes neighbouring v_i in the graph. This is a slight modification to the definition of symbolic representation with circuits given by Balcazar et al.

A similar result as that in [5] is then proved on this model of symbolic representation. More precisely, if graph problem A is hard for complexity class C under *covered* log time reductions, then sA is hard for the complexity class exponentially higher than C under polynomial time reductions. The key difference in this result is that the reductions which are required for this theorem to hold are now *covered* log time reductions. We will now explain what these are, noting that since this reduction f is a reduction to a *graph* problem A, then for an arbitrary binary string $x, f(x)$ is a graph. Thus, a covered log time reduction (f) is a log time reduction with the additional property that for all binary strings x, the degree of any vertex in $f(x)$ is at most $O(\log(|x|))$.

Therefore, this paper successfully adapts the proof presented by Balcazar et al for a symbolic model which is more related to graph searching in AI. This is important for our research, as it gives us further ideas on how the proof from Balcazar et al can be modified for different symbolic representations such as SDDs.

Research Problem 2

[28] K. Chatterjee and M. Henzinger, “Efficient and Dynamic Algorithms for Alternating Büchi Games and Maximal End-Component Decomposition,” J. ACM, vol. 61, no. 3, p. 15:1-15:40, Jun. 2014, doi: 10.1145/2597631.

This paper investigates alternating reachability games, which are defined as a graph $G = (V, E)$ with a partition of the vertex set $V = V_1 \cup V_2$ and a subset $B \subset V$ of goal vertices. A token is placed at a given initial vertex. Now, if the token is on a vertex $v_1 \in V_1$ then player 1 moves the token along an edge outgoing from v_1, and if the token is on a vertex $v_2 \in V_2$ then player 2 moves the token along an edge outgoing from v_2. The canonical problem for a reachability game is to determine whether the first player can “win” by reaching a goal state.

The key finding of this paper is an $O(n^2)$ algorithm (where n is the number of vertices in G) for solving an alternating reachability game, which is a strict improvement on the previously best-known result $O(mn)$ (where m is the number of edges in G). Furthermore, Catherjee et al are the first to formulate a dynamic data structure for storing an alternating reachability game, which can determine the winner in $O(1)$ time at the cost of requiring $O(n)$ time for each edge/vertex insertion/deletion.

Despite this, the relevance of this paper to our second research problem is not due to its key results, but rather due to its clear presentation of known results and applications. More specifically, this paper provides a list of applications of solving alternating reachability games, including the verification of safeness and liveness properties in open systems, thereby, motivating our second research question. The paper also states known results, including the P-TIME completeness of finding the winner of an alternating reachability game, thereby, allowing us to postulate that the succinct AFA emptiness problem will be EXPTIME complete.

[35] M. Valdats, “Descriptional and Computational Complexity of the Circuit Representation of Finite Automata,” in Language and Automata Theory and Applications, Cham, 2018, pp. 105–117. doi: 10.1007/978-3-319-77313-1_8.

In this paper Valdats investigates the BC-complexity of deterministic finite automata (DFAs), which intuitively represents the complexity of the transition function of an automaton.

More formally, given a DFA A we can enumerate the symbols in its alphabet in binary using $\log_2(\Sigma)$ bits for each symbol. We can also enumerate its states in binary using $\log_2\left(Q \right)$ bits for each state. We then say that a Boolean circuit C represents A when: $C\left(p, a\right) = q$ (note that p, a, q are written in binary) if and only if $\delta\left(p, a\right) = q$. The BC-complexity of an automata A is therefore defined as

$$BC(A) = \min\{\ C \ \mid C\ represents\ A\}$$

where the size of a circuit is the number of gates it contains.

This paper proves the key result, which is a characterisation of the BC-complexity of an automaton A in terms of its state complexity s. In particular it is found that

$$BC(A) \sim \frac{s}{\log(s)}$$

In the latter half of the paper, it is then defined that given a problem on DFAs (A), the equivalent succinct problem is $sA = \{circuits\ C \mid the\ automata\ represented\ by\ C \in A\}$. Valdats then finds that the succinct DFA emptiness problem, the succinct DFA equivalence problem and the succinct DFA state reachability problem are all PSPACE-complete, whereas their non-succinct counterparts are NL-complete. This is proved using the generalisations given by Balcazar et al [5]. In particular, the original DFA problems are shown to be NL-complete under *log time reductions*, allowing the author to conclude that the succinct problems are therefore PSPACE-complete.

This paper is useful for our second research problem, as it gives a method for us to solve our problem. In particular, to prove that the succinct Boolean circuit non-emptiness problem on AFAs is EXPTIME-hard, we simply need to show that the non-emptiness problem on explicit AFAs is PTIME-hard under log time reductions.

[36] M. Y. Vardi, "An automata-theoretic approach to linear temporal logic," in Logics for Concurrency: Structure versus Automata, F. Moller and G. Birtwistle, Eds., in Lecture Notes in Computer Science. Berlin, Heidelberg: Springer, 1996, pp. 238–266. doi: 10.1007/3-540-60915-6_6.

The problem of program verification takes as input: a program - which is often modelled as a Kripke Structure (or another graph like structure), and a specification - which is often defined with a linear temporal logic formula, and determines whether the program obeys the specification. More details of this problem can be found in [26].

Vardi's main contribution in this paper is giving a method to solve program verification with automata. More precisely, given a linear temporal logic formula ϕ, one can construct an alternating Buchi automaton A with alphabet $\Sigma = \{assignments\ on\ variables\ in\ \phi\}$. The words accepted by A will therefore represent a series of assignments of variables, or in other words, computations. Vardi provides an algorithm to build the automaton A in exponential time, such that $L(A)$ is exactly the satisfying computations of ϕ. Thus, the problem of program verification can be solved by determining whether the computations of the program are a subset of $L(A)$.

However, the usefulness of this paper doesn't lie in the main result, but rather in the Lemmas which are used to prove this main result. In particular, constructions are given to build an alternating automaton A given two alternating automata A_1 and A_2 such that $L(A) = L(A_1) \cap L(A_2)$, or $L(A) = L(A_1) \cup L(A_2)$. We can therefore adapt these constructions to work for automata which are represented symbolically as well, allowing us to use these adapted constructions in our proofs for research problem 2.

[37] M. Holzer and M. Kutrib, "Descriptional and computational complexity of finite automata—A survey," Information and Computation, vol. 209, no. 3, pp. 456–470, Mar. 2011, doi: 10.1016/j.ic.2010.11.013.

This paper is a survey of known complexity results regarding problems on explicitly represented finite automata. The problems of membership, emptiness, universality and equivalence are explored on DFAs, Nondeterministic Finite Automata (NFAs) and AFAs. The complexity of each of these problems is reported, and their proofs are provided with reference to the original papers. For instance, the universality problem is NL-complete for DFAs, and PSPACE-complete for NFAs and AFAs, unless the language is unary, in which case it is L-complete for DFAs, coNP-complete for NFAs and PSPACE-complete for AFAs.

Thus, even though the paper does not make any new contributions, it is extremely useful for our approach to the second research problem. This is because we can firstly examine the existing reduction in the proof of PTIME-hardness for AFA emptiness to see if it is an LT reduction. If it is, we can apply the result from [5] to conclude that succinct Boolean circuit AFA emptiness is EXPTIME-hard. Secondly, if this fails, we can adapt the hardness proof and prove the EXPTIME-hardness more tediously instead.

References

[1] "Welcome to Core." https://www.core.edu.au/ (accessed Mar. 28, 2023).

[2] "Conference Ranks." http://www.conferenceranks.com/#data (accessed Mar. 28, 2023).

[3] "Acceptance rates of Theoretical Computer Science Conferences." https://www.lamsade.dauphine.fr/~sikora/ratio/confs.php (accessed Mar. 28, 2023).

[4] A. Darwiche, "SDD: A New Canonical Representation of Propositional Knowledge Bases".

[5] J. L. Balcázar, A. Lozano, and J. Torán, "The Complexity of Algorithmic Problems on Succinct Instances," in *Computer Science: Research and Applications*, R. Baeza-Yates and U. Manber, Eds., Boston, MA: Springer US, 1992, pp. 351–377. doi: 10.1007/978-1-4615-3422-8_30.

[6] H. Veith, "Languages represented by Boolean formulas," *Inf. Process. Lett.*, vol. 63, no. 5, pp. 251–256, Sep. 1997, doi: 10.1016/S0020-0190(97)00134-8.

[7] H. Veith, "How to encode a logical structure by an OBDD," in *Proceedings. Thirteenth Annual IEEE Conference on Computational Complexity (Formerly: Structure in Complexity Theory Conference) (Cat. No.98CB36247)*, Jun. 1998, pp. 122–131. doi: 10.1109/CCC.1998.694598.

[8] G. Kovásznai, H. Veith, A. Fröhlich, and A. Biere, "On the Complexity of Symbolic Verification and Decision Problems in Bit-Vector Logic," in *Mathematical Foundations of Computer Science 2014*, E. Csuhaj-Varjú, M. Dietzfelbinger, and Z. Ésik, Eds., in Lecture Notes in Computer Science. Berlin, Heidelberg: Springer, 2014, pp. 481–492. doi: 10.1007/978-3-662-44465-8_41.

[9] D. Sawitzki, "The Complexity of Problems on Implicitly Represented Inputs," in *SOFSEM 2006: Theory and Practice of Computer Science*, J. Wiedermann, G. Tel, J. Pokorný, M. Bieliková, and J. Štuller, Eds., in Lecture Notes in Computer Science. Berlin, Heidelberg: Springer, 2006, pp. 471–482. doi: 10.1007/11611257_45.

[10] H. Galperin and A. Wigderson, "Succinct representations of graphs," *Inf. Control*, vol. 56, no. 3, pp. 183–198, Mar. 1983, doi: 10.1016/S0019-9958(83)80004-7.

[11] C. H. Papadimitriou and M. Yannakakis, "A note on succinct representations of graphs," *Inf. Control*, vol. 71, no. 3, pp. 181–185, Dec. 1986, doi: 10.1016/S0019-9958(86)80009-2.

[12] B. Das, P. Scharpfenecker, and J. Torán, “CNF and DNF succinct graph encodings,” *Inf. Comput.*, vol. 253, pp. 436–447, Apr. 2017, doi: 10.1016/j.ic.2016.06.009.
[13] JoséL. Balcázar, “The complexity of searching implicit graphs,” *Artif. Intell.*, vol. 86, no. 1, pp. 171–188, Sep. 1996, doi: 10.1016/0004-3702(96)00014-8.
[14] T. Bylander, “The computational complexity of propositional STRIPS planning,” *Artif. Intell.*, vol. 69, no. 1, pp. 165–204, Sep. 1994, doi: 10.1016/0004-3702(94)90081-7.
[15] K. W. Wagner, “The complexity of combinatorial problems with succinct input representation,” *Acta Inform.*, vol. 23, no. 3, pp. 325–356, Jun. 1986, doi: 10.1007/BF00289117.
[16] T. Lengauer and K. W. Wagner, “The correlation between the complexities of the nonhierarchical and hierarchical versions of graph problems,” *J. Comput. Syst. Sci.*, vol. 44, no. 1, pp. 63–93, Feb. 1992, doi: 10.1016/0022-0000(92)90004-3.
[17] J. Feigenbaum, S. Kannan, M. Y. Vardi, and M. Viswanathan, “Complexity of problems on graphs represented as OBDDs,” in *STACS 98*, M. Morvan, C. Meinel, and D. Krob, Eds., in Lecture Notes in Computer Science, vol. 1373. Berlin, Heidelberg: Springer Berlin Heidelberg, 1998, pp. 216–226. doi: 10.1007/BFb0028563.
[18] P. Scharpfenecker, “Bounded-Depth Succinct Encodings and the Structure they Imply on Graphs,” *Theory Comput. Syst.*, vol. 62, no. 5, pp. 1125–1143, Jul. 2018, doi: 10.1007/s00224-017-9787-4.
[19] B. Bollig and M. Buttkus, “On the Relative Succinctness of Sentential Decision Diagrams.” arXiv, Feb. 13, 2018. Accessed: Mar. 29, 2023. [Online]. Available: http://arxiv.org/abs/1802.04544
[20] A. Lozano and J. L. Balcázar, “The complexity of graph problems for succinctly represented graphs,” in *Graph-Theoretic Concepts in Computer Science*, M. Nagl, Ed., in Lecture Notes in Computer Science. Berlin, Heidelberg: Springer, 1990, pp. 277–286. doi: 10.1007/3-540-52292-1_20.
[21] M. Faella and G. Parlato, *Reachability Games Modulo Theories with a Bounded Safety Player **. 2022.
[22] L. P. Kaelbling and T. Lozano-Pérez, “Hierarchical task and motion planning in the now,” in *2011 IEEE International Conference on Robotics and Automation*, May 2011, pp. 1470–1477. doi: 10.1109/ICRA.2011.5980391.
[23] Bryant, “Graph-Based Algorithms for Boolean Function Manipulation,” *IEEE Trans. Comput.*, vol. C–35, no. 8, pp. 677–691, Aug. 1986, doi: 10.1109/TC.1986.1676819.
[24] J. R. Burch, E. M. Clarke, K. L. McMillan, D. L. Dill, and L. J. Hwang, “Symbolic model checking: 1020 States and beyond,” *Inf. Comput.*, vol. 98, no. 2, pp. 142–170, Jun. 1992, doi: 10.1016/0890-5401(92)90017-A.
[25] R. P. Kurshan, “The complexity of verification,” in *Proceedings of the twenty-sixth annual ACM symposium on Theory of Computing*, in STOC ’94. New York, NY, USA: Association for Computing Machinery, May 1994, pp. 365–371. doi: 10.1145/195058.195194.
[26] E. M. Clarke, T. A. Henzinger, H. Veith, and R. Bloem, Eds., *Handbook of Model Checking*. Cham: Springer International Publishing, 2018. doi: 10.1007/978-3-319-10575-8.

[27] L. Vinkhuijzen and A. Laarman, “Symbolic Model Checking with Sentential Decision Diagrams,” 2020, pp. 124–142. doi: 10.1007/978-3-030-62822-2_8.

[28] K. Chatterjee and M. Henzinger, “Efficient and Dynamic Algorithms for Alternating Büchi Games and Maximal End-Component Decomposition,” *J. ACM*, vol. 61, no. 3, p. 15:1-15:40, Jun. 2014, doi: 10.1145/2597631.

[29] A. Pnueli and R. Rosner, “On the synthesis of a reactive module,” in *Proceedings of the 16th ACM SIGPLAN-SIGACT symposium on Principles of programming languages*, in POPL ’89. New York, NY, USA: Association for Computing Machinery, Jan. 1989, pp. 179–190. doi: 10.1145/75277.75293.

[30] P. Ramadge and W. Wonham, “Supervisory control of a class of discrete event systems,” *SIAM J. Control Optim.*, vol. 25, pp. 206–230, Jan. 1987, doi: 10.1137/0325013.

[31] A. Mahanti and A. Bagchi, “AND/OR graph heuristic search methods,” *J. ACM*, vol. 32, no. 1, pp. 28–51, Jan. 1985, doi: 10.1145/2455.2459.

[32] L. de Alfaro and T. A. Henzinger, “Interface automata,” *ACM SIGSOFT Softw. Eng. Notes*, vol. 26, no. 5, pp. 109–120, Sep. 2001, doi: 10.1145/503271.503226.

[33] “On the menbership problem for functional and multivalued dependencies in relational databases | ACM Transactions on Database Systems.” https://dl.acm.org/doi/10.1145/320613.320614 (accessed Mar. 29, 2023).

[34] “Descriptive Complexity.” https://people.cs.umass.edu/~immerman/book/descriptiveComplexity.html (accessed Mar. 29, 2023).

[35] M. Valdats, “Descriptional and Computational Complexity of the Circuit Representation of Finite Automata,” in *Language and Automata Theory and Applications*, S. T. Klein, C. Martín-Vide, and D. Shapira, Eds., in Lecture Notes in Computer Science. Cham: Springer International Publishing, 2018, pp. 105–117. doi: 10.1007/978-3-319-77313-1_8.

[36] M. Y. Vardi, “An automata-theoretic approach to linear temporal logic,” in *Logics for Concurrency: Structure versus Automata*, F. Moller and G. Birtwistle, Eds., in Lecture Notes in Computer Science. Berlin, Heidelberg: Springer, 1996, pp. 238–266. doi: 10.1007/3-540-60915-6_6.

[37] M. Holzer and M. Kutrib, “Descriptional and computational complexity of finite automata—A survey,” *Inf. Comput.*, vol. 209, no. 3, pp. 456–470, Mar. 2011, doi: 10.1016/j.ic.2010.11.013.