Inferentialism as a Research Paradigm in Computer Science Education: a Review and Theoretical Contribution

J. V. Crossley[a]
*and Marjahan Begum[b] and Jo Wood[b]
[ab] Department of Computer Science,
City St Georges,
University of London, UK

December 4, 2025

Abstract

Background and Context: Research in computer science education has been shaped by paradigms such as constructivism, socio-constructivism, and pragmatism. While influential, these approaches have limits in explaining inferential claims and argumentation. Inferentialism offers a way to integrate both individual and collaborative reasoning by situating meaning-making within shared normative practices.

Objective: This paper provides a conceptual review of research paradigms in computer science education, situating them within wider educational theory and highlighting their strengths and limitations. Against this background, it introduces inferentialism—an established philosophical theory of semantics—as a novel paradigm with potential for computing education.

Methods: Taking the literature on inferentialism in education as a starting point, we conducted a narrative, critical review of literature on research paradigms in computer science education. To illustrate inferentialism's value for computing, we present synthetic examples grounded in empirical data.

Findings: Inferentialism offers a structured way to analyze reasoning, revealing latent misconceptions and reframing them as context-sensitive expressions of normative engagement. It complements existing paradigms by systematically linking commitments, entitlements, and normative rules in both monologic and dialogic contexts.

Implications: Adopting inferentialism can enrich assessment and feedback, support curriculum design, and inspire new methodologies for conducting qualitative data analysis. It also opens avenues for empirical studies and the development of AI-driven educational tools grounded in inferentialist principles.

Inferentialism; computer science education; research paradigms; student argumentation; assessment; deontic scorekeeping.

*CONTACT J. Crossley Email: julia.crossley.2@citystgeorges.ac.uk

1 Introduction

A number of philosophical positions are used in educational analysis. These exist on a spectrum and at one end we have paradigms such as positivism, at the other socio-constructivism. Other stances lie somewhere in between those two extremes; examples are constructivism and pragmatism. Each comes with caveats that are worsened in light of AI's and automation's growing role in education (Raihan, Siddiq, Santos, & Zampieri, 2025; Sarsa, Denny, Hellas, & Leinonen, 2022; Tenenberg & Malmi, 2022).
Positivism seeks objective insights through empirical generalization (Alexander, 2013). However, it overlooks individual learning experiences and imposes objectivity on inherently subjective processes; an example of this is in automatically generated feedback that lacks sensitivity to learning context (Tenenberg & Malmi, 2022).
Constructivism embraces this subjectivity but, in doing so, can sacrifice the importance of established standards in contributing to shared meaning, which is necessary in feedback giving; in the advent of AI it also removes oversight on what the student is learning (Raihan et al., 2025). Socio-constructivism, more radical than its predecessor, rejects the existence of individual learning, seeing it entirely as discursive.
Pragmatism overlooks the importance of the process of learning, as it focuses primarily on outcomes, and process is an essential aspect of constructing meaning; this is exacerbated by the convenience of using LLMs for learning (Raihan et al., 2025; Sarsa et al., 2022).
We suggest another philosophical position that has shown some promise in Mathematics education (Derry, 2017; Drimalla, 2025; Noorloos, Taylor, Bakker, & Derry, 2014; Patel & Pfannkuch, 2025; Radford, 2017; Ruben Noorloos, Taylor, Bakker, & Derry, 2017; Taylor, Noorloos, & Bakker, 2017): the theory of inferentialism, a philosophical position sharing features with constructivism. Inferentialism views conceptual understanding as a network of concepts linked by inferential relations, shaped through norm governed social practices.
This **position paper** introduces this theory, reviews its application in mathematics education. It proposes an original extension to computing education, a field where it remains largely unexplored. Focusing on written data as a key mode of assessment in higher education, the paper suggests a new research paradigm to motivate future research. It does this using carefully designed synthetic illustrative examples, and outlines directions for future research.

2 Materials and methods

2.1 Materials

2.1.1 Literature selection

We conducted a structured narrative review. Our literature selection fell into three categories of literature:

1. Inferentialism in education, including the foundational literature on the philosophical theory;
2. Reviews of the research paradigms present in computer science education. We carried out searches in the ACM Digital Library, Computer Science Education and Informatics in Education, with no limits on dates in the first instance. We used keywords such as "research paradigm", "theory", "conceptual" combined with "review".
3. Reviews of the application of dominant research paradigms in computing education.
4. Qualitative studies analyzing academic student data.

Inclusion criteria We included publications that were peer-reviewed in journals and conferences.

Exclusion criteria We excluded purely philosophical treatments of inferentialism without reference to education, apart from the foundational literature and opinion pieces. We also excluded papers on the application of teaching tools and research paradigms suited to purely quantitative, un-interpretive studies. The reason for this is explained in the literature review.

Selection process Our selection process began with seminal works on inferentialism in education, identified through recommendations from leading scholars in the field and citation chaining. This literature raised our interest in applying inferentialism to computer science education and is the underlying motivator for this paper. We then examined paradigmatic literature in computer science education, focusing on constructivism, socio-constructivism, and pragmatism. Finally, we searched for studies on reasoning patterns in CSEd to identify potential sites where inferentialist analysis could apply. No prior work was identified that applied inferentialism directly to computer science contexts.

2.1.2 Illustrative examples

We used examples provided as synthetic demonstrations, designed to illustrate how inferentialist analysis could be applied in practice. They are not intended as empirical validation, but as conceptual tools to concretize abstract principles and motivate further study. They were designed using insights gained from the analysis of real data, literature on misconceptions in computer science and literature on effective methods of assessment. They therefore contain problem statements that allow for granular analysis; the responses contain patterns we observed in real student responses and common misconceptions according to the literature. While they are not verbatim reproductions, they are faithful analogues of authentic student responses, applied to a problem statement similar to the ones in real data.

2.2 Methods

As this is a position paper, our methods differ from those of an empirical study. We do not present data collection; instead we use illustrative demonstrations to concretize inferentialist principles in computer science education contexts.
Our approach has three strands:

1. **Literature review** The literature review, conducted iteratively, draws on seminal works in inferentialism from its philosophical basis and mathematics education. It is then contrasted with the dominant research paradigms, providing points of contrast.

2. **Theorization** This section situates inferentialism in computing education, giving examples that aid the reader in understanding the principles before they are used in analysis. It then discusses how the analysis can be applied to monologic data in computing education.

3. **Synthetic examples** Constructed examples are presented, inspired by real data (exam scripts, coursework and studies documented in literature). These bring out the pedagogical value of the analysis, offering insights on how to apply it to authentic responses, while ensuring anonymity.

The purpose of this approach is to conceptually illustrate how inferentialist analysis can reveal patterns that are particular to the learning context and situate them within a context where the student's skills can be measured.

2.2.1 Design of illustrative examples

We constructed synthetic examples that presented with features such as reasoning steps, misconceptions and ambiguities we observed in real data. These examples were developed based on three sources: anonymized patterns noticed in exam scripts and coursework from undergraduate modules in algorithms, programming, and theory of computation; findings from the literature on known student misconceptions in these domains; findings from literature on assessments that best bring out computational skills.
These were designed to preserve the characteristic features of authentic student responses within the context of the assessments. For algorithms, they incorporated misunderstandings around tree balancing, traversal strategies, optimization, appropriate choice of structure and runtime analysis. In programming, syntactically correct code was created that presented with semantic flaws, reflecting misconceptions in preserving immutability, mishandling references and poor consideration of how to combine program requirements.
In theoretical computer science, attempts at mathematical proofs were constructed to illustrate common misapplications of proof principles, such as conflating necessary and sufficient conditions, and content such as the pumping lemma.
By designing examples that presented with incoherencies at a content level as well as a broader skill level, we present illustrative examples that give rich material for inferentialist analysis. These examples

provide opportunities to demonstrate how inferentialist principles can bring out subtleties in the data that less granular analysis cannot.

2.3 Predominant research paradigms in computer science education

2.3.1 What is a research paradigm?

Philosophies, research paradigms, learning theories and methodologies are loosely defined. This is perhaps because they often overlap. Our concerns are first and foremost to present the philosophical theory of inferentialism, and frame it as a methodological system (a research paradigm) that can then be used to derive appropriate methodologies for conducting research in computing education. This framing has been explored in mathematics education (Bakker & Derry, 2011; Bakker & Hußmann, 2017; Derry, 2017; Drimalla, 2025) and we will use those insights to present inferentialism in a manner that makes it applicable to methodological design.

To this end, we discuss the predominant research paradigms in computing education, suggesting how adopting inferentialism could fill gaps in methodologies and empirical findings. We therefore define *research paradigm* as a system that guides how knowledge is defined and interpreted. They sit within a philosophical theory; they guide the design of methodologies and determine criteria for research. They are distinct from learning theories in that they offer a *lens for interpretation* rather than a description of learning processes.

The most influential *distinct* research paradigms in computer science are constructivism and socio-constructivism (Malmi et al., 2014; Sinclair, Malmi, Sheard, Kinnunen, & Simon, 2019). Pluralism is also documented as a dominant system; however it is loosely defined, which is unsurprising given its nature. Pragmatism and positivism are both philosophical theories and research paradigms. In practice, they often appear within pluralistic approaches, especially given the dominance of intervention-based and quantitative research in computing education (Heckman, Carver, Sherriff, & Al-zubidy, 2021). Although these have their place in research, we are not proposing methodological adoptions that fall in that category. We will therefore restrict our counterparts to constructivism and socio-constructivism. In what follows, we describe how these two paradigms differ from inferentialism, highlighting what the latter can offer in filling gaps in methodology and interpretation in computer science education.

2.3.2 Research paradigms vs pedagogical frameworks

We propose inferentialism as operating at the level of a research paradigm, shaping how we interpret meaning-making and reasoning. Frameworks such as SOLO taxonomy (Biggs & Collis, 1982), PRIMM (Sentance, Waite, & Kallia, 2019), or cognitive apprenticeship (Guzdial, 1994) can be situated within various paradigms, even though they align well with constructivist based ones; they provide specific methodological or pedagogical tools. Our contribution is not to compete with those frameworks but to suggest inferentialism as a broader interpretive stance that can enrich them and contextualize them to a specific learning environment.

2.3.3 Constructivist based paradigms in Computer Science Education

A particular emphasis on active learning and collaboration has emerged in the last few decades, not only in instructional approaches but also in theory; this is evident in the adoption of constructivist and socio-contructivist methodologies in designing new frameworks and suggesting pedagogical interventions (Malmi et al., 2014; Sinclair et al., 2019). Frameworks such as PRIMM (Sentance et al., 2019), the Block model (Schulte, 2008), TIPSEE (Salac, Thomas, Butler, Sanchez, & Franklin, 2020) and Cognitive Apprenticeship in computer science education (Guzdial, 1994), are examples of this. These theories all sit well under a constructivist/socioconstructivist philosophy, in that they all rest on principles of active participation.

The same can be said of empirical studies exploring student experience (C. Lewis, Yasuhara, & Anderson, 2019; Lishinski & Yadav, 2021; Ojha, West, & Lewis, 2024); or research into the benefits of collaborative learning (Batten & Ross, 2021; Fong, Huang, Alawini, Silva, & Herman, 2024; Hawlitschek, Berndt, & Schulz, 2022; Porter et al., 2016; Zingaro & Porter, 2014) . These all present academic success in computer science as highly dependent on personal and social experience. In turn, research indicates that taking a collaborative approach to pedagogy can have positive outcomes for student retention, engagement and academic results (Batten & Ross, 2021; Porter et al., 2016).

Conversely, some research indicates problems with this approach. Alternative studies indicate that the

benefits of collaborative activities, such as pair-programming, are unrelated, if not negatively correlated, with academic outcomes (Bowman, Dorn, Rodger, McGill, & Ericson, 2020, 2021). The features of a constructivist and socio-constructivist approach, such as minimal guidance, can lead to less effective learning and untackled misconceptions due to untrained working memory and loose mental models (Ben-Ari, 1998; Kirschner, Sweller, & Clark, 2006).
Research advocating for socio-constructivism in particular stresses that an essential aspect of its application is the analysis of data, such as participation and practice, rather than mental models only (Tenenberg & Knobelsdorf, 2014). This requires making a careful distinction between the learning of a community and the learning of the individual within a community: constructivism and socio-constructivism cannot do this together pluralistically for the same piece of data. One can triangulate interpretations of the same data using both perspectives, but this is not the same as employing a single lens that sees both individual and social learning.
This is where inferentialism becomes powerful. Such a lens brings awareness of the context of learning, clearly tracking where inconsistencies in argumentation appear, how they are mitigated and how consistencies are justified. Such a lens is inferentialism.
More precisely, we are not presenting the principles of inferentialism in the context of teaching tools, such as PRIMM or SOLO. Its application lies in the dimension of data interpretation. Where frameworks guide teaching or assessment design, inferentialism provides a lens for analyzing the data embedded in student work. We will demonstrate in more detail how inferentialism offers avenues for doing this in a way that it is systematic.

2.4 Inferentialism as complementary to Constructivist stances

2.4.1 What Is Inferentialism?

Inferentialism is a philosohical theory of semantics established by Robert Brandom (Brandom, 1994, 2000a). It holds that the *meaning* of a concept is shaped by the inferential links it has to other concepts, forming a network of concepts; and that these inferential links are built through engagement with contextual norms. To understand a concept one must understand what it implies, what can be deduced from it, what contradicts it, in the context the concept is learnt (Brandom, 1994, 2000b; Peregrin, 2006). It is primarily a theory of *expression*: in its own context it is also a theory of meaning, as according to inferentialism those two notions cannot be separated. Network semantics are mediated through three principal features: normative rules; inferential commitments and entitlements; deontic scorekeeping. This will be discussed further in subsection 2.7.
It is grounded in pragmatism, in that it is concerned with how knowledge is externalized; but its view of learning is in the construction of coherent knowledge, rather than practical outcomes(Brandom, 1994).
Constructivism has similarities to inferentialism: it views learning as context dependent (Noorloos et al., 2014). However, it views concepts as existing in isolated 'mental structures', independently of any environment. Socio-constructivism, a variant of constructivism that focuses on the social aspect of learning, does not integrate personal context with normative context (Ruben Noorloos et al., 2017).
Inferentialism aims to shift this view of concepts to how they are used in normative contexts, and integrate them with the student's interpretation of norms. It avoids reliance on assumptions on internal mental states, making it more appropriate for interpreting what students actually express (Noorloos et al., 2014; Ruben Noorloos et al., 2017). It is more interpretive than positivism, which treats knowledge as immutable (Uegatani & Otani, 2021); more analytical than constructivism and socio-constructivism, in that it offers a systematic way to interpret what both positions view as context dependent (Noorloos et al., 2014); more focused on building coherent knowledge than pragmatism, which values practical application without requiring justification (Brandom, 1994).

2.4.2 Inferentialism in Mathematics Education

Inferentialism has been offered as an alternative to constructivism and socio-constructivism by educational researchers in mathematics (Noorloos et al., 2014; Ruben Noorloos et al., 2017). Inferentialism does not negate these two positions; its focus is on learning through networks of concepts, linked together by inferences. This implicitly means that a concept cannot be reified (Sfard, 1991) on its own, it needs other concepts to link to.
Its application to maths has focused on primary education. The data given in these studies are transcriptions of spoken data and drawings during the collection of this data. The focus of the analysis is on the linguistic exchange and active knowledge construction rather than expression: the student(s) is

learning during the process of the data collection.
In particular Drimalla's study looks at a group activity, focusing on deontic scorekeeping of commitments and entitlements amongst the participants during the activity. They engage in a game of asking questions, such as *why* and *what if*, using each others commitments as entitlements. One particular student demonstrates mastery in this context by refuting claims that are not consistent with the normative rules of the context. The student demonstrates further mastery by making valid inferences beyond other participants commitments (Drimalla, 2025).
Seidouvi and Schindler use a similar context: they track the inferences made in a collaborative activity. The deontic score-keeping consists of tracking the claims participants make, how these are used by any participant and whether it is consistent with the normative rules of the context (Seidouvy & Schindler, 2020). Normative rules in this context are set up by the content and how students interpret the content: for example a student mentions a norm about the precision needed in measurement tasks (Seidouvy & Schindler, 2020). These studies characterize the learning context as "social", and to a certain extent we can interpret this as *spoken* during the process of the scorekeeping.
As such, studies applying inferentialism to mathematics education particularly emphasize tracking commitments and entitlements; and the pedagogical value of students justifying their inferential claims, using the normative rules of the context.
Our discussion of inferentialism will focus for the most part on its usefulness for analyzing argumentation in computing education contexts. However, we are not suggesting its adoption in a loose sense. Our analysis is grounded in a coherent philosophical framework, while remaining accessible and adaptable to the practical concerns of computing education research.

2.5 Inferentialism in Computer Science Education

Having situated inferentialism in the philosophical landscape of educational analysis, we outline why examining educational analysis from this perspective is beneficial to CSEd.
We argue that not enough current research integrates insights into student learning experience and comprehension, and that an analytical perspective that operates on multiple dimensions is needed. This includes examining how content is *taught* (at the institutional, lecture module and educator level) and how it is *learned* (at the student level), thereby using these insights when analyzing student data.
A one dimensional approach to analysis can present in other areas of pedagogy. Educators often take a pragmatic approach to what students output and this impacts how they assess students; their process, and by consequence how they constructed their knowledge and how coherent it is is not assessed (Schellekens et al., 2021). This tension between process and product in educational research is enduring (Sfard, 1998); inferentialism is valued by its proponents because it frames learning as the result of both applying inferential principles and engaging in reflective thought (Taylor et al., 2017). This means recognizing both shared conceptual understandings across a subject and the pedagogical culture shaped by institutions, educators, and students' subjective experiences.
Inferentialism has the potential to offer such a framework, accommodating variation in how meaning is expressed, including through natural language or pseudo-code.
We start by describing inferentialism in more depth. It is a philosophical position that holds that meaning is constructed through shared reasoning about concepts. Although it originates in the philosophy of language, its relevance to education has grown and this has been demonstrated in mathematics education. There is also growing interest in the CSEd community in research focusing on student reasoning and argumentation (Begum et al., 2025; Kallia, 2023; Kallia, Cutts, & Looker, 2022; C. M. Lewis, 2023). We argue that applying inferentialism may yield more specific insights and may complement this area of research.

2.5.1 Relevance to Education

Careful analysis of how meaning is constructed is especially important in written work, where reasoning must be inferred from what is 'on paper'. This can empower educators and researchers to:

- identify misconceptions as context-specific misunderstandings, offering deeper insight into applying of concepts;
- provide feedback on latent patterns of misconception missed by pragmatic or positivist approaches;
- surface patterns inspiring research and supporting frameworks that guide active learning.

To ground this analysis, we draw on Brandom's inferentialism: content is a network of concepts linked by rules that govern incompatibilities and consequences (Brandom, 1994; Rahman & Redmond, 2013). Reasoning can then be analyzed through the inferential relations between claims; the validity of these inferential relations are determined by underlying contextual rules (normative rules) (Rahman & Redmond, 2013).
Educational researchers and practitioners can make use of these normative rules by examining how students apply them (Noorloos et al., 2014). This can raise interesting questions about how students engage with content and how context, such as style of assessment, shapes their application of concepts within the norms of teaching and learning.

2.6 From Theory to Practice

Applying inferentialism to education can present with some issues around the nature of knowledge. Inferentialism treats understanding of concepts as constructed through experience, rather than fixed and universal. This requirement can be difficult to interpret in an educational setting where concepts have a clear shared meaning, independent of any social context, and any assessment requires using this shared meaning as a standard by which to judge students' work.
In particular, where there is little interaction and dialogue, such as a written non-collaborative assessment, how can we interpret experiential understanding? Through the norms the educator creates in their teaching environment.
How can we interpret these norms in a domain such as mathematics or computer science, where meaning is fixed? Through the network of concepts taught to the student; through teaching and learning devices used to convey information, such as notional machines and analogies. We discuss this in more detail in subsection 2.8.

2.7 Inferentialist features applied to education

2.7.1 Normative rules

Normative rules are standards used to judge reasoning. In education, they define what is considered scientifically sound or plausible within a knowledge system. These rules are not isolated facts but reflect how concepts relate - what contradicts, supports, or follows from what. They are shaped by the teaching context: curriculum content, delivery, and conceptual connections.
Although content may be treated as objectively true, its framing is influenced by pedagogy. Applying inferentialism in education requires articulating these context specific rules. For example, the idea that *a recursion without a base case will not terminate* is both a factual statement and normative standard in early undergraduate teaching -students use it to assess correctness within that context. We will refer to the set of normative rules and the network of concepts as the *normative context*.

2.7.2 Inferential commitments and entitlements

We can treat inferential entitlements as the foundational concepts that are explicitly presented in the module content; those concepts educators can assume students understand and are entitled to use. Building on these concepts, students make inferential commitments: the conclusions they derive from making inferences, which they can then invoke as premises in subsequent reasoning steps.
An example of a commitment is the assertion *this language violates the pumping lemma for context free languages*; if it is justified it can then be used as an entitlement to assert *this language is not context free*. Whether this entitlement holds within the teaching context depends on whether it respects the relevant normative rules. In any case it can be taken as a valid entitlement within the context of the student solving the problem.
At times inferential commitments are hidden, in that they are consequences of an inferential claim rather than the conclusion of one. Choosing an implementation of a structure entails making inferential commitments: these commitments are in a hidden layer of abstraction and because of this are implicit.
An implementation of the abstract data type Set might print its members in a different order if implemented using a hash set, as opposed to a tree set:

```
Set<Integer> firstSet = new HashSet<>();
firstSet.add(5);
firstSet.add(14);
firstSet.add(3);
```

```
...
firstSet.add(7);

Set<Integer> secondSet = new TreeSet<>();
secondSet.add(5);
secondSet.add(14);
secondSet.add(3);
...
secondSet.add(7);
```

Although either implementation may be acceptable for a wider problem, in a situation where for instance access to set elements has the precedence of order of insertion, a set implement using a a TreeSet will not return the desired value. This is an example of where the student may not have fully considered the inferential commitments they are implicitly making by choosing an implementation.
Other 'tacit' inferential commitments may appear in between normative contexts. An example of this is where the student needs to express themselves computationally in more than one language, such as Java and plain english.
The following shows a simple implementation of Peterson's algorithm in Java, followed by a justification for why it ensures mutual exclusivity. The Java code is correct; the worded justification is not. Although there is clear familiarity with the code, the worded description neither describes it properly nor explains how mutual exclusion is ensured.

```
class Peterson{
    private volatile boolean[] flag = new boolean[2];
    private volatile int turn;

    public void lock(int i){
        int j = 1- i;
        flag[i] = true;
        turn = j;
        while(flag[j] && turn==j){
        }
    }

    public void unlock(int i){
        flag[i] = false;
    }
}
```

What happens in the code is that two different threads are created. They enter the critical section at the same time. The variable turn gets assigned to one of them and so it is always guaranteed that if one is running the other one isn't. This means the two threads are mutually exclusive.

If we consider the two descriptions separately, one is correct according to the normative rules of the context: concurrent programs in Java. The other is not, as the worded description is not enough to justify correctness. It also does not describe the code accurately. To be able to do this effectively, the student would need to understand the 'tacit' inferential commitments they have made by writing the code down and the ones that are inherent to the problem statement.

2.7.3 Deontic score-keeping

This is the process of tracking students' inferential commitments and entitlements; specifically whether these align with the normative rules of the given context.
It involves evaluating whether a claim based on an entitlement leads to a commitment that is valid within the established rules.
Deontic score-keeping can also aid identifying inconsistencies in student reasoning. Such inconsistencies may not necessarily indicate incorrect assertions in a Platonic sense, but rather reveal the structure of the student's reasoning. This recasts misconceptions as insights into students normative engagement with content, rather than deviations from a fixed model.
An example of a normative rule violation in Java identified through deontic score-keeping is:

- Entitlement: *a reference p2 is initially set to the same object as p1; it is then reassigned to a new object.*
- Commitment: *both p1 and p2 refer to the same object.*
- Normative rule violation: reassigning a reference does not change the other references to the original object.

Figure 1 illustrates the deontic score-keeping process. Claim p, a reliable fact, is an entitlement; it is used as a premise in an inference. An inferential link leads to conclusion q; this inferential link either respects or violates normative rules and commitments. In turn q becomes a commitment. If it is used in a subsequent inference it is also an entitlement.
An aspect of assessment which deontic score-keeping can bring out is sequential misapplications of normative rules resulting in a correct commitments.
Consider the following example. The student is given this exercise:

Write a function that takes an undirected graph (as an adjacency list of vertices with unique integer values). For each vertex with an even value, double the value of each of its neighbors. No vertex should be doubled more than once. Return the updated graph. Find a suitable example to illustrate your function is correct.

The student proceeds to write the following function:

```
function doubleAdjacentValues(graph):
    visited = new Set()
    for vertex in graph:
        if vertex.value is even:
            for neighbor in vertex.adjacent:
                if neighbor.value not in visited:
                    neighbor.value = neighbour.value * 2
                    visited.add(neighbor.value)
    return graph
```

The student goes on to give an example of a graph where none of the data values are doubles of each other, even though this restriction is not stated in the problem statement. As a result the function *appears* to be correct. It isn't for any graph that doesn't fit this restriction: if one vertex is doubled and its new value matches the one of another vertex, this second vertex may be incorrectly skipped. This is an example of how incorrect assumptions result in a plausibly correct outcome, which can be better identified through deontic score-keeping.

2.8 Inferentialism for different pedagogical purposes

We now discuss how to apply inferentialist principles to monologic data. Application to dialogic data is covered in the literature in Mathematics Education (Bakker & Derry, 2011; Drimalla, 2025; Seidouvy & Schindler, 2020); we suggest that Inferentialism could be similarly applied to a CS context, such as during a pair programming assessment.
We are also interested in how to apply it to monologic data. Taylor et al contribute a theoretically based framework for integrating cognitive and social dimensions of learning using inferentialism (Taylor et al., 2017), and some of these insights could be used to construct an approach to applying deontic score-keeping to monologic data.
In particular, knowledge can be seen as the internalization of a socio-normative process (Taylor et al., 2017); and monologic discourse can be seen as an internalized dialogue involving the management of entitlements and commitments.
By aligning the analysis with the specific teaching and learning context in which the data was collected - and by using deontic score-keeping to examine the students' inferential moves - we can remain sensitive to the normative dimension of their reasoning. This avoids the imposition of a Platonic interpretation of knowledge, which, as Uegatani (Uegatani & Otani, 2021) argues, is a feature of any retrospective analysis and consequently incompatible with Inferentialism.
We would therefore like to suggest an approach for retrospective analysis in CSEd contexts, which we do in the next section.

In doing so we do not intend to classify learning into tiered stages as Bloom (Bloom, Engelhart, Furst, Hill, & Krathwohl, 1956), SOLO (Biggs & Collis, 1982), APOS (Dubinsky & McDonald, 2001) or Sfard (Sfard, 1991); but rather to analyse the progress and coherence of students' argumentation (Drimalla, 2025) in a manner that is consistent with computer science norms.The outcome can be either feedback to the student or insights for research purposes. In the following section we introduce this approach for the application of inferentialism to a Computer Science context, and we then illustrate with an example in algorithm design.

2.8.1 Applying Inferentialism to Monologic Data

Here we justify how monologic discourse could be analyzed using deontic-scorekeeping and normative rules. When someone speaks or writes, they are externalizing their reasoning. This is known to have pedagogical value, even if there no explicit interaction; learning by 'teaching' is valuable, as is any form of practice of a skill. The person interacts with concepts and normative rules through the learning process, and externalizing their thoughts yields data that can be analyzed using the same principles as dialogic data.
Brandom implies this as well: making assertions results in the 'availability' of the assertions for making other assertions (Brandom, 1994). In a dialogic situation, this involves a game of giving and asking for reasons (Brandom, 1994, 2000a). In a monologic situation, where it is expected claims will be justified (such as in an undergraduate assessment), the same should occur. If it does not, tracking commitments and entitlements using inferentialist principles can reveal how well a person can question their own claims using normative rules. It can also situate their knowledge relative to the norms, via feedback. It is therefore useful to analyze inferential moves in monologic data, adopting the position of an 'implicit' challenger of the inferential claims the interlocutor makes.

3 Results

In keeping with the nature of this position paper, the results are demonstrations rather than empirical findings. Their function is to show the potential analytical strength of inferentialism when applied to computer science education data.
They can be seen a conceptual exemplification: they give an indication of what inferentialism could uncover, but do not measure prevalence or generalizability in authentic cohorts. Future empirical studies are needed for that purpose.

3.1 A motivational example

Consider statements p and q:

" p: An AVL tree T has n nodes. "
" q: The runtime of a search in T is $\mathcal{O}(\log_2 n)$. "

An incomplete reason one might give for why "there is p, so I infer q" could be: "A binary tree must have a height of $\log_2 n$, so would have a worst case runtime of $\mathcal{O}(\log_2 n)$ for any search in it. " The only evidence that is clearly used to support the claim "there is p, so I infer q" is that the structure is binary tree based and that it has n nodes. A complete explanation, where understanding is explicit, would give reference to the specific attributes of the structure given in claim p (an AVL tree) and how they imply q:

- the height is proportional to $\log_2 n$ because the structure is self-balancing;
- the search has a runtime of $\mathcal{O}(\log_2 n)$ because the tree is ordered; this means redundant paths can be eliminated at every node that is searched and so the full depth would only be traversed once.

At first glance, this detailed explanation may seem unnecessary. Whether the student made the connection or was asked to justify it, the reasoning on why it is correct is plausible if we assume the student meant 'AVL tree' rather than 'binary tree', has a height that 'is close to $\log_2 n$' rather than 'is $\log_2 n$' and that the student has enough knowledge to know the implications of using an AVL tree on runtime. However, in tasks where a student may have to justify or defend the correctness of an approach that relies on understanding of the methods and constraints on a particular structure, being explicit and clear becomes being correct - which quickly becomes *solving the problem you were asked to solve.*

3.2 Suggested approach for application to CS

The following is an outline of how the features of inferentialism can be applied to a computer science teaching and learning context. We emphasize that this is not a formal framework, but a proposed approach intended to stimulate discussion and exploration within the CSEd community. Features given in section 2.6 are detailed, explaining how to apply it to an educational context; where necessary details of how to apply it to a computer science context are given. These can be found in table 1. We now present examples to illustrate our position.

3.3 Synthetic example as illustrative demonstrations

Analyzing argumentative coherence requires detailed data from a individual participants. Ethical considerations prevent us from describing a single source of data at the level of granularity required to illustrate inferentialist techniques. We have therefore decided to use synthetic examples that embody characteristic features we observed in real exam scripts and coursework responses. Examples of these characteristics are incoherent choice of data structures, inconsistent implementation, and poorly applied proof principles. In this way, we preserve the pedagogical and analytical value of real data while ensuring ethical integrity and protecting participants' anonymity.

3.3.1 Deontic Scorekeeping for a worded description of an Algorithm

Areas of the curriculum such as algorithms are prone to misconceptions in topics such as optimization, greedy algorithms and dynamic programming (Schindler, Hußmann, Nilsson, & Bakker, 2017; Tilanterä, Sorva, Seppälä, & Korhonen, 2024; Velázquez-Iturbide, 2025); and foundational content such as data structures and complexity (M. et al., 2023; Nelson et al., 2020; Zingaro et al., 2018). We feel this exercise serves as an illustrative example of how an inferentialist approach can aid in unpicking student intent and offering valuable feedback.

Problem statement: Given a collection of n items, each with a unique name and an access count, describe an algorithm that efficiently handles the following operations in $O(\log_2 n)$ runtime: adding a new item, removing an existing item, and retrieving the item with the k^{th} largest count. Provide an explanation of how the algorithm meets these runtime requirements for each operation, clearly stating any implied constraints on the problem.

Illustrative response We use a synthetic response as an example, constructed to reflect common patterns in real data. This example was constructed based on recurring patterns observed in anonymized student submissions from a second-year algorithms module.
It was constructed based on recurring patterns observed in anonymized student submissions for a second year data structures and algorithms module; a fourth year advanced algorithm modules, over two consecutive cohorts. Common misconceptions included misunderstandings and misapplication of tree balancing, traversal methods and runtime implications. Other more subtle misconceptions were in what students judged as information *hiding* (Colburn & Shute, 2007) rather than a misapplication of principles. This led to incoherent and insufficiently described responses. These features could be better emphasized through deontic score-keeping.
This example illustrates core inferentialist principles and how deontic score-keeping reveals normative rule violations in monologic responses. It also serves as a prototype to illustrate empirical applications of the approach.

- *I would use a binary search tree, so that I can update the structure and so that it has the right runtime.*
- *If we added an item it would have count* 0 *so we would traverse the tree until it is in the right place.*
- *To manage this collection I would use a binary search tree. Each node would hold as data the count of how many times it has been retrieved; this count would be updated each time a node is accessed.*
- *To add an item, I would insert the node in the binary search tree and place it in the correct position to maintain the structure. I would do this by doing an in-order traversal to find the first semi-leaf node and add it to the empty child node.*

- *To remove an existing item, I would do a search to find the node corresponding to the item and remove it from the tree. If it has a left child, I would promote the left child to the position the removed node was in. If it has a right child only, I would promote it to the position of the removed node.*
- *To retrieve the item with the k^{th} largest count, I would search through the tree using an in-order traversal. When I find the node that has k as its count I would retrieve it and update the count held at the node. I would keep track of the visited nodes using a stack and then count to the right number in the sequence.*
- *This has a runtime of $\mathcal{O}(\log_2 n)$ because of the maximum depth of the tree. It is correct because it adds, removes and searches for the required nodes.*

Deontic scorekeeping Table 2 shows the deontic score-keeping for this hypothetical response. By examining the normative rule violations, we can see that the 'student' has misconceptions about the structure of binary trees; the specifications of an augmented tree needed to satisfy the runtime requirement; standard methods on binary tree based structures, and as a consequence how they can be appropriately used to solve the problem. As a result they cannot clearly argue:

- why the management of the collection can be done in $\mathcal{O}(\log_2 n)$ time;
- how conditions for implicit restructuring ensures the algorithm correctly manages the collection;
- how an appropriate traversal of the structure could be used to retrieve the k^{th} most popular item without traversing the whole structure;
- why the choice of a binary structure that is ordered and self-balancing yields an optimal algorithm.

These are all features that can be more easily uncovered by closely examining how coherent the student's decisions are and whether they result in a cohesive argument.

3.3.2 Complementary examples

Here we present a complementary example to illustrate deontic scorekeeping applied to normative contexts where underlying semantic rules differ from those of everyday language, such as executable code. We also present an example of applying deontic scorekeeping to a normative context where the semantic rules are implicit, such as proof in theory of computation.

Deontic scorekeeping for executable code The coherence of an argument in solving a problem is not only in the individual inferential claims made, such as lines of code and conditions. It is also in the ordering of the arguments, such as the order of code execution.
This is an additional normative rule that does not appear in a worded description. We therefore suggest that in applying inferentialist techniques to programming code, deontic scorekeeping follows the order the code is executed in. The importance of control flow is therefore embedded in any empirical insights or feedback to the student.
In this next example we choose code that is syntactically correct but semantically flawed. The flaws would escape both dynamic and static tools; the example demonstrates how issues can be uncovered by deontic scorekeeping where traditional big detection may fall short.

Problem statement Implement an immutable class representing a fraction. Ensure that equivalent fractions generate the same value and include standard arithmetic operations on them.

Illustrative response

```
public final class Fraction {
    private final int[] rep;
    public Fraction(int num, int den) {
        if (den == 0) throw new IllegalArgumentException("den=0");
        this.rep = new int[] { num, den };
    }
    public int[] getRep() {
```

```
        return rep;
    }
    public int numerator()   { return rep[0]; }
    public int denominator() { return rep[1]; }

    public Fraction add(Fraction other){
        int n = this.rep[0] * other.rep[1] + other.rep[0] * this.rep[1];
        int d = this.rep[1] * other.rep[1];
        return new Fraction(n, d);
    }
    public Fraction subtract(Fraction other){
        int n = this.rep[0] * other.rep[1] - other.rep[0] * this.rep[1];
        int d = this.rep[1] * other.rep[1];
        return new Fraction(n, d);
    }
    public Fraction multiply(Fraction other){
        int n = this.rep[0] * other.rep[0];
        int d = this.rep[1] * other.rep[1];
        return new Fraction(n, d);
    }
    public Fraction divide(Fraction other){
        if (other.rep[0] == 0) throw new ArithmeticException("divby0");
        int n = this.rep[0] * other.rep[1];
        int d = this.rep[1] * other.rep[0];
        return new Fraction(n, d);
    }
    @Override public boolean equals(Object obj){
        if (!(obj instanceof Fraction f)) return false;
        return this.rep[0] == f.rep[0] && this.rep[1] == f.rep[1];
    }
    @Override public int hashCode() {
        return 31 * rep[0] + rep[1];
    }
    @Override public String toString() {
        return rep[0] + "/" + rep[1];
    }
}
```

Deontic scorekeeping This is an example where the errors are in fulfilling the implied specifications of the problem. Here two requirements are violated, one stated in the problem and the other more implicit.
The immutability requirement is violated by the getter getRep(), which allows access to a field that has a mutable type. It returns a reference that is exposed to mutation by external code. The implied commitment due to rule violation is therefore that the field could be changed via another reference.
An implicit requirement that is not tackled is fraction equivalence; this is another rule violation. Implied commitments as a result of this rule violation are consequences for any external uses of the class, such as in lookups or hash-based collections
A way to tackle both requirements could have been to add normalization condition to the constructor, such that it occurs before final field assignment. A latent feature that could be extracted from this insight is that the student does not fully consider the specifications and does not combine them.

Deontic scorekeeping for an exercise in Theory of Computation Computer Science students often present with misconceptions when the use of formal mathematics is required; this is particularly the case in areas of the curriculum focusing heavily on mathematical proof, such as Theory of Computation (Frede & Knobelsdorf, 2018). Groundwork for understanding the notation and standard tools, such as proof by induction, is needed (del Vado Vírseda, 2021). In turn this can lead to responses where certain steps are consistent with normative rules, in a manner that may suggest rote learning; but the overarching aim of the proof is lost. There may be implicit inferential commitments they are not aware of.
Conversely, the student may make an inferential claim that is logically correct but inconsistent with the standard approach they claim to be using: in this sense they are violating norms that are 'local' to the problem statement's context.
We therefore present an example to illustrate how deontic scorekeeping can be used to uncover exactly which inferential claims violate the normative rules of the context.

Problem statement Show that $L = \{ww^R | w \in \{a, b, c\}^*\}$ is a context-free languages.

Illustrative response *If the language is context-free, it will satisfy the pumping lemma for context-free languages. We could write a string in L as $s = xayaz$. This means that for length $p = |xyz| \leq s$ we can have:*

- $|aa| \geq 1$;
- $|aya| \leq p$;
- $xa^iya^iz \in L \ \forall i \geq 0$

This means that L satisfies the pumping lemma for context-free languages, so the language is context-free.

Deontic scorekeeping In this case the normative rules include those of standard mathematical proofs and the module content. This means understanding how the principles of the Pumping Lemma could be applied to a specific regular language; how it can be used in a proof by contradiction; and how it would *have to* be used in a direct proof.
In this particular example the student attempts to apply the pumping lemma for context-free languages to a direct proof. One violated rule is that the pumping lemma can only be used to prove that a language is not context-free; a shallow interpretation of this misconception could be simply that the student has confused a necessary condition with a sufficient one. However, careful examination of the normative context, namely the knowledge that all context-free languages satisfy the pumping lemma for context-free languages, allows for a more nuanced interpretation.
The norm that

$$L \textit{ is context-free} \implies L \textit{ satisfies the pumping lemma}$$

is conflated with

$$L \textit{ satisfies the pumping lemma} \implies L \textit{ is context-free.}$$

Similarly, the normative rule that the hypothetical value of the pumping length should determine the expression for the strings in the language, rather than the other way around, is violated.
In other words

$$\textit{for } L \textit{ there is } p > 0 \implies \textit{there are strings } s \textit{ in } L \textit{ such that}$$

is conflated with

$$\textit{there are strings } s \textit{ in } L \textit{ such that} \implies \textit{there is } p > 0.$$

In both cases, the student does not apply entitlements and commitments in the logical order according to the normative context. The order of reasoning is therefore reversed. This is a latent feature that can be picked up through tracking inferential claims and commitments.

4 Discussion

As with the previous sections, this discussion reflects the nature of a position paper, exploring the potential of inferentialism in computer science education contexts.

4.1 Limitations in the data presented

The illustrative examples are synthetic and we have designed them for the purpose of demonstrating the potential for applying inferentialism to computer science education. They closely mirror authentic student argumentation in real data and allowed us to clearly show how inferential argumentation, as exemplified by entitlements, commitments and normative rules, can aid in analyzing data.
The description of an algorithm gave potential for ambiguity, which made our example rich in demonstrating how deontic scorekeeping can uncover latent rule violations. Similarly, the example using executable code displayed how 'hidden' inferential commitments can impact the reliability of a solution. The example involving a proof demonstrated how argumentative steps could appear logically viable in isolation

while concurrently violating normative rules that are closer to the context of the assessment, in this case applying the pumping lemma to a context-free language.
We focused on monologic, written data, reflecting the anonymous nature of summative assessments in higher education. In such settings, educators must rely solely on what is expressed in writing or code. We felt that this made developing an approach to help interpret how students communicate their reasoning in writing all the more valuable.
However, we do acknowledge that the example does not demonstrate programming language specific norms; this is another facet of the development of the conceptual framework that needs development. In such cases, the platonic interpretation of code correctness by machines may obscure the student's intended meaning and this is something that should be kept in mind when assessing the students' use of norms. We also acknowledge that the application of the methodology to a real-life situation may be less straightforward. The data may be less suitable, particularly if the assessment is closed. Categorizations of reasoning may yield lower inter-rater agreement, which could undermine the aim of the analysis, if inter-rater agreement is required. Finally, we also feel that applying the approach to a broader range of CS content could yield more insights on the effectiveness of the methodology we are developing. Future work will explore these directions.
There are also limitations in the inferentialist approach. Up until now we have not discussed them. The reason for this is that we aim to start a discussion on how this interesting alternative to constructivism (Noorloos et al., 2014) can be effectively applied to a variety of areas of the CS curriculum and a variety of modes of assessment. More research is needed in this area, which is out of the scope of this motivational paper.

4.2 Practical implications of adopting inferentialism in computer science education

4.2.1 Implications for educators of computer science

The potential uses of inferentialist principles to practitioners is wide. We have framed our examples in manner that allows educators to use deontic scorekeeping to:

- Pinpoint where in the data misconceptions become apparent.
- Reframe mistakes as engagement with the normative learning context, to better judge how the student has fared. This in turn can contribute to more thoughtful teaching and feedback.
- Enable practionners to reflect on their practice, by considering how their students make use of the normative learning context and how this compares to broader undergraduate computer science norms.
- Design tasks and assessments that encourage the student to make their inferential commitments explicit (Brandom, 1994); to articulate reasons for their inferential claims (Brandom, 2000b); to careful tacit commitments when they make an inferential claim.

4.2.2 Implications for researchers in computer science education

The implications for researchers is the potential development of methodologies that allow for systematic analysis of qualitative data within normative teaching contexts. In particular, it can provide guidance on acknowledging not only the positionality of the researcher, but the positionality of the data. In deep qualitative analysis it is important to consider not only the broader context the data was gathered in (such as the demographic of students, the type of assessment, the length of the lecture module) but also the specifics of the teaching environment (such as conventions used to teach the content, seen and unseen assessments). By considering the norms, researchers can uncover deeper qualitative insights on their data and offer increased transparency in their approach.

4.2.3 Implications for researchers across fields

Although the subject has not been discussed in this paper yet, inferentialism offers potential to influence research methodologies outside of computer science education.
It offers a mindset for analyzing the normative coherence of qualitative data, not only within educational analysis. Domains that rely on argumentation such as law, ethics or politics can use inferentialist principles to assess the coherence of argumentation. In the social sciences, inferentialism's emphasis on

contextualized norms can help researchers evaluate how situational factors shape sociological phenomena. The same can be said of assessment of academic papers, as these invariably rely heavily on claim justification and appropriate situating of the normative context.
Inferentialism therefore offers avenues for developing systematic approaches to analyzing qualitative data, in contexts where consistency with 'local' rules are important.

4.2.4 Implications for artificial intelligence

Inferentialism and artificial intelligence already have a cross-sectional area of a research: this area is primarily focused around defining formal logics for the theory, such a proof-theoretic semantics; philosophical debates on its applicability to artificial intelligence; models that 'learn' from deontic scorekeeping.
We are not aware of any research delving into the operationalization of deontic scorekeeping according to 'local' norms; we think this is an area that has potential as well, particularly for educational research and practice.

4.2.5 Implications for philosophy of semantics/education

Inferentialism has primarily been discussed in the context of dialogic exchange. Here we have applied it to monologic data, arguing that the student has a dialogue with themselves in justifying claims. This is mediated by their awareness of the norms of the content; the nature of assessment; the problem statement; and the knowledge that it will be assessed, such that the assessor can play the role of a 'silent' counterpart.

5 Conclusion

As a position paper, our aim is not to present empirical findings but to stimulate reflection and provide a conceptual tool for future research. We encourage empirical studies that apply inferentialist analysis to authentic student data to both test empirical validity and extend the ideas presented in this paper.
Inferentialism offers a promising and, to our knowledge, unexplored perspective for deeper analysis in CSEd. By emphasizing the contextual and normative dimensions of meaning-making, inferentialism enables a more nuanced interpretation of student work. This paper suggests an approach grounded in inferentialist principles and demonstrates its potential. It illustrates how deontic score-keeping can surface latent misconceptions and reasoning. Despite our examples being synthetic, they reflect real-world ambiguities and challenges in effectively assessing students. We acknowledge that further work is needed to consider the applicability of the approach to diverse CS curricula, assessments and programming languages. Nonetheless, we believe this approach holds promise for enhancing feedback, informing curriculum design, and guiding future empirical research.
We see opportunities for investigating how it can be applied to dialogic data, such as during pair programming tasks or research into team dynamics, such as Kallia et al.'s work in collective argumentation (Kallia et al., 2022). Such work could be elaborated through deontic score-keeping, for example by tracing how team members make use of each other's inferential commitments and how they negotiate inferential incompatibility.
In monologic contexts, such as those explored in Kallia et al's study on individual reasoning (Kallia, 2023), Oliveira et al.'s analysis of think aloud sessions (Oliveira, Keuning, & Jeuring, 2023), Nijenhuis-Voogt et al.'s study on application of algorithmic concepts to unseen content (Nijenhuis-Voogt, Bayram-Jacobs, Meijer, & Barendsen, 2022) and Lewis' work on code comprehension strategies (C. M. Lewis, 2023), through the lens of deontic score-keeping could further insights into how students arrive at conclusions. For example, a study such as Nijenhuis-Voogt et al.'s (Nijenhuis-Voogt et al., 2022) could be extended by applying deontic score-keeping to uncover patterns in how students identify and use relevant inferential entitlements within unfamiliar material.
We invite the CSEd community to engage with these ideas and consider how inferentialism might inform both best practices and future research. We offer this new approach as a conceptual tool to provoke reflection and exploration, not as a prescriptive framework. Its value lies in its potential to surface reasoning patterns and inform future research. Potential directions include empirical investigations into student reasoning trajectories, the development of assessment tools incorporating deontic score-keeping, and intervention studies grounded in inferentialist principles. We hope this serves as stimulus for further collaborations in this area.

Acknowledgments

We would like to thank Dr Cristina Gacek and Dr Panos Giannopoulos at City St Georges, University of London for letting us use the data we based our synthetic examples on.

Declaration of interest statement

None.

References

Alexander, H. (2013). Traditions of inquiry in education: Engaging the paradigms of educational research. In *A companion to research in education* (pp. 13–25). Dordrecht, Netherlands: Springer. Retrieved from https://link.springer.com/chapter/10.1007/978-94-007-6809-3_2

Bakker, A., & Derry, J. (2011). Lessons from inferentialism for statistics education. *Mathematical Thinking and Learning*, *13*(1–2), 5–26. Retrieved from https://doi.org/10.1080/10986065.2011.538293

Bakker, A., & Hußmann, S. (2017). Inferentialism in mathematics education: Introduction to a special issue. *Mathematics Education Research Journal*, *29*(4), 395–401. Retrieved from https://link.springer.com/article/10.1007/s13394-017-0224-4

Batten, J. S., & Ross, M. S. (2021). A systematic review of social constructivist pedagogies in computing and their effects on broadening participation for women in undergraduate computing. In *Proceedings of the american society for engineering education (asee) annual conference & exposition.* American Society for Engineering Education.

Begum, M., Crossley, J., Strömbäck, F., Akrida, E., Alpizar-Chacon, I., Evans, A., ... Thorgeirsson, S. (2025). A pedagogical framework for developing abstraction skills. In *2024 working group reports on innovation and technology in computer science education* (p. 258–299). New York, NY, USA: Association for Computing Machinery. Retrieved from https://doi.org/10.1145/3689187.3709613

Ben-Ari, M. (1998, Mar). Constructivism in computer science education. *ACM SIGCSE Bulletin*, *30*(1), 257–261.

Biggs, J. B., & Collis, K. F. (1982). *Evaluating the quality of learning: The solo taxonomy (structure of the observed learning outcome)evaluating the quality of learning: The solo taxonomy (structure of the observed learning outcome).* New York: Academic Press.

Bloom, B., Engelhart, M., Furst, E., Hill, W., & Krathwohl, D. (1956). *Taxonomy of educational objectives: The classification of educational goals. handbook i: Cognitive domain.* New York: David McKay Company.

Bowman, N., Dorn, B., Rodger, S. H., McGill, M. M., & Ericson, B. J. (2020). Pair programming in computer science education: A multi-institutional study. *Journal of Women and Minorities in Science and Engineering*, *26*(3), 211–232.

Bowman, N., Dorn, B., Rodger, S. H., McGill, M. M., & Ericson, B. J. (2021). A multi-institutional study of pair programming in introductory computing courses. *ACM Transactions on Computing Education*, *21*(4), 1–25.

Brandom, R. (1994). *Making it explicit: Reasoning, representing, and discursive commitment.* Cambridge, MA: Harvard University Press.

Brandom, R. (2000a). *Articulating reasons: An introduction to inferentialism.* Cambridge, MA: Harvard University Press.

Brandom, R. (2000b). Articulating reasons: An introduction to inferentialism. *Philosophy and Phenomenological Research*, *60*(3), 611–614.

Colburn, T., & Shute, G. (2007). Abstraction in computer science. *Minds and Machines*, *17*(2), 169–184. Retrieved from https://link.springer.com/article/10.1007/s11023-007-9061-7

del Vado Vírseda, R. (2021). Learning from the impossible: Introducing theoretical computer science in cs mathematics courses. In *Proceedings of the 52nd acm technical symposium on computer science education* (p. 952–958). New York, NY, USA: Association for Computing Machinery. Retrieved from https://doi.org/10.1145/3408877.3432475

Derry, J. (2017). An introduction to inferentialism in mathematics education. *Mathematics Education Research Journal*, *29*(4), 403–418. Retrieved from https://link.springer.com/article/10.1007/s13394-017-0193-7

Drimalla, J. (2025). Inferences, mastery, and objectivity: examining student understanding of mathematical concepts with inferentialism. *Educational Studies in Mathematics*, *119*, 393-410.

Dubinsky, E., & McDonald, M. (2001). Apos: A constructivist theory of learning in undergraduate mathematics education research. In D. Holton, M. Artigue, U. Kirchgräber, J. Hillel, M. Niss, & A. Schoenfeld (Eds.), *The teaching and learning of mathematics at university level* (Vol. 7, pp. 275–282). Dordrecht: Springer.

Fong, M. M., Huang, S., Alawini, A., Silva, M., & Herman, G. L. (2024). Exploring computing students' sense of belonging before and after a collaborative learning course. In *Proceedings of the 55th acm technical symposium on computer science education v. 1* (p. 359–365). New York, NY, USA: Association for Computing Machinery. Retrieved from https://doi.org/10.1145/3626252.3630850

Frede, C., & Knobelsdorf, M. (2018). Exploring how students perform in a theory of computation course using final exam and homework assignments data. In *Proceedings of the 2018 acm conference on international computing education research* (p. 241–249). New York, NY, USA: Association for Computing Machinery. Retrieved from https://doi.org/10.1145/3230977.3230996

Guzdial, M. (1994). Software-realized scaffolding to facilitate programming for science learning. *Interactive Learning Environments*, *4*(1), 1–44. Retrieved from https://doi.org/10.1080/1049482940040101

Hawlitschek, A., Berndt, S., & Schulz, S. (2022). Empirical research on pair programming in higher education: A literature review. *Computer Science Education*, *33*(3), 400–428.

Heckman, S., Carver, J. C., Sherriff, M., & Al-zubidy, A. (2021, October). A systematic literature review of empiricism and norms of reporting in computing education research literature. *ACM Trans. Comput. Educ.*, *22*(1). Retrieved from https://doi.org/10.1145/3470652

Kallia, M. (2023). The search for meaning: Inferential strategic reading comprehension in programming. In *Proceedings of the 2023 acm conference on international computing education research - volume 1* (p. 1–14). New York, NY, USA: Association for Computing Machinery. Retrieved from https://doi.org/10.1145/3568813.3600135

Kallia, M., Cutts, Q., & Looker, N. (2022). When rhetorical logic meets programming: Collective argumentative reasoning in problem-solving in programming. In *Proceedings of the 2022 acm conference on international computing education research - volume 1* (p. 120–134). New York, NY, USA: Association for Computing Machinery. Retrieved from https://doi.org/10.1145/3501385.3543975

Kirschner, P. A., Sweller, J., & Clark, R. E. (2006). Why minimal guidance during instruction does not work: An analysis of the failure of constructivist, discovery, problem-based, experiential, and inquiry-based teaching. *Educational Psychologist*, *41*(2), 75–86.

Lewis, C., Yasuhara, K., & Anderson, R. (2019). Feeling like a computer scientist: The impact of belonging and identity in cs education. In *Proceedings of the 15th international computing education research conference (icer).*

Lewis, C. M. (2023). Examples of unsuccessful use of code comprehension strategies: A resource for developing code comprehension pedagogy. In *Proceedings of the 2023 acm conference on international computing education research - volume 1* (p. 15–28). New York, NY, USA: Association for Computing Machinery. Retrieved from https://doi.org/10.1145/3568813.3600116

Lishinski, A., & Yadav, A. (2021, June). Self-evaluation interventions: Impact on self-efficacy and performance in introductory programming. *ACM Trans. Comput. Educ.*, *21*(3). Retrieved from https://doi.org/10.1145/3447378

M., B., Haglund, P., A., K., V., L., M., M., F., S., & A., T. (2023). Empirical evaluation of a differentiated assessment of data structures: The role of prerequisite skills. *Informatics in Education*, *23*, 57-99.

Malmi, L., Sheard, J., Simon, Bednarik, R., Helminen, J., Kinnunen, P., ... Taherkhani, A. (2014). Theoretical underpinnings of computing education research – what is the evidence? In *Proceedings of the tenth annual conference on international computing education research (icer '14)* (pp. 27–36). ACM.

Nelson, G. L., Strömbäck, F., Korhonen, A., Begum, M., Blamey, B., Jin, K. H., ... Monga, M. (2020). Differentiated assessments for advanced courses that reveal issues with prerequisite skills: A design investigation. In *Proceedings of the working group reports on innovation and technology in computer science education* (p. 75–129). New York, NY, USA: Association for Computing Machinery. Retrieved from https://doi.org/10.1145/3437800.3439204

Nijenhuis-Voogt, J., Bayram-Jacobs, D., Meijer, P. C., & Barendsen, E. (2022). Recognizing algorithmic concepts in new contexts: An analysis of students' reasoning. *Informatics in Education*, *21*(3), 541–568. Retrieved from https://doi.org/10.15388/infedu.2022.16

Noorloos, R., Taylor, S., Bakker, A., & Derry, J. (2014). An inferentialist alternative to constructivism in mathematics education. In *Proceedings of the joint meeting of pme 38 and pme-na 36* (pp. 321–328). Vancouver, Canada: PME 38 Proceedings. Retrieved from https://files.eric.ed.gov/fulltext/ED599976.pdf

Ojha, V., West, L., & Lewis, C. M. (2024). Computing self-efficacy in undergraduate students: A multi-institutional and intersectional analysis. In *Proceedings of the 55th acm technical symposium on computer science education v. 1* (p. 993–999). New York, NY, USA: Association for Computing Machinery. Retrieved from https://doi.org/10.1145/3626252.3630811

Oliveira, E., Keuning, H., & Jeuring, J. (2023). Student code refactoring misconceptions. In *Proceedings of the 2023 conference on innovation and technology in computer science education v. 1* (p. 19–25). New York, NY, USA: Association for Computing Machinery. Retrieved from https://doi.org/

10.1145/3587102.3588840

Patel, A., & Pfannkuch, M. (2025). An inferentialism-based framework for capturing statistical concept formation over time. *Statistics Education Research Journal*, *24*(1), 3. Retrieved from https://iase-pub.org/ojs/SERJ/article/view/714

Peregrin, J. (2006). Inferentialism: Why rules matter. *Philosophia*, *34*(3), 311–329.

Porter, L., Bouvier, D., Cutts, Q., Grissom, S., Lee, C., McCartney, R., ... Simon, B. (2016). A multi-institutional study of peer instruction in introductory computing. In *Proceedings of the 47th acm technical symposium on computing science education* (p. 358–363). New York, NY, USA: Association for Computing Machinery. Retrieved from https://doi.org/10.1145/2839509.2844642

Radford, L. (2017). On inferentialism. *Mathematics Education Research Journal*, *29*, 493 - 508.

Rahman, S., & Redmond, J. (Eds.). (2013). *Logic, argumentation & reasoning: Interdisciplinary perspectives from the humanities and social sciences*. Dordrecht: Springer.

Raihan, N., Siddiq, M. L., Santos, J., & Zampieri, M. (2025). Large language models in computer science education: A systematic literature review. In *Proceedings of the 56th acm technical symposium on computer science education v. 1* (p. 938–944). New York, NY, USA: Association for Computing Machinery. Retrieved from https://doi.org/10.1145/3641554.3701863

Ruben Noorloos, R., Taylor, S., Bakker, A., & Derry, J. (2017). Inferentialism as an alternative to socioconstructivism in mathematics education. *Mathematics Education Research Journal*, *29*, 437-453. Retrieved from https://link.springer.com/article/10.1007/s13394-017-0189-3

Salac, J., Thomas, C., Butler, C., Sanchez, A., & Franklin, D. (2020). Tippsee: A learning strategy to guide students through use - modify scratch activities. In *Proceedings of the 51st acm technical symposium on computer science education* (p. 79–85). New York, NY, USA: Association for Computing Machinery. Retrieved from https://doi.org/10.1145/3328778.3366821

Sarsa, S., Denny, P., Hellas, A., & Leinonen, J. (2022). Automatic generation of programming exercises and code explanations using large language models. In *Proceedings of the 2022 acm conference on international computing education research - volume 1* (p. 27–43). New York, NY, USA: Association for Computing Machinery. Retrieved from https://doi.org/10.1145/3501385.3543957

Schellekens, L. H., Bok, H. G., de Jong, L. H., van der Schaaf, M. F., Kremer, W. D., & van der Vleuten, C. P. (2021). A scoping review on the notions of assessment as learning (aal), assessment for learning (afl), and assessment of learning (aol). *Studies in Educational Evaluation*, *71*, 101094. Retrieved from https://doi.org/10.1016/j.stueduc.2021.101094

Schindler, M., Hußmann, S., Nilsson, P., & Bakker, A. (2017). Sixth-grade students' reasoning on the order relation of integers as influenced by prior experience: an inferentialist analysis. *Mathematics Education Research Journal*, *29*(2), 165–186.

Schulte, C. (2008). Block model: an educational model of program comprehension as a tool for a scholarly approach to teaching. In *Proceedings of the fourth international workshop on computing education research* (p. 149–160). New York, NY, USA: Association for Computing Machinery. Retrieved from https://doi.org/10.1145/1404520.1404535

Seidouvy, A., & Schindler, M. (2020). An inferentialist account of students' collaboration in mathematics education. *Mathematics Education Research Journal*, *32*(3), 411–431.

Sentance, S., Waite, J., & Kallia, M. (2019). Teaching computer programming with primm: a sociocultural perspective. *Computer Science Education*, *29*(2-3), 136–176.

Sfard, A. (1991). On the dual nature of mathematical conceptions: Reflections on processes and objects as different sides of the same coin. *Educational Studies in Mathematics*, *22*, 1-36.

Sfard, A. (1998). On two metaphors for learning and the dangers of choosing just one. *Educational Researcher*, *27*(2), 4–13.

Sinclair, J., Malmi, L., Sheard, J., Kinnunen, P., & Simon. (2019). Computing education theories: What are they and how are they used? *ACM Transactions on Computing Education (TOCE)*, *19*(3), 1–35.

Taylor, S. D., Noorloos, R., & Bakker, A. (2017). Mastering as an inferentialist alternative to the acquisition and participation metaphors for learning. *Journal of Philosophy of Education*, *51*(4), 768–784.

Tenenberg, J., & Knobelsdorf, M. (2014). Out of our minds: a review of sociocultural cognition theory. *Computer Science Education*, *24*(1), 1–24. Retrieved from https://www.tandfonline.com/doi/10.1080/08993408.2013.869396

Tenenberg, J., & Malmi, L. (2022). Editorial: Conceptualizing and using theory in computing education research. *ACM Transactions on Computing Education (TOCE)*, *22*(4), 38:1–38:8.

Tilanterä, A., Sorva, J., Seppälä, O., & Korhonen, A. (2024). Students struggle with concepts in dijkstra's

algorithm. In *Proceedings of the 2024 acm conference on international computing education research - volume 1* (p. 154–165). New York, NY, USA: Association for Computing Machinery. Retrieved from https://doi.org/10.1145/3632620.3671096

Uegatani, Y., & Otani, H. (2021). A new ontology of reasons for inferentialism: redefining the notion of conceptualization and proposing an observer effect on assessment. *Mathematics Education Research Journal*, *33*, 183–199. Retrieved from https://link.springer.com/article/10.1007/s13394-019-00289-8

Velázquez-Iturbide, (2025). Reflections on teaching algorithm courses. In *Proceedings of the 56th acm technical symposium on computer science education v. 1* (p. 1148–1154). New York, NY, USA: Association for Computing Machinery. Retrieved from https://doi.org/10.1145/3641554.3701937

Zingaro, D., & Porter, L. (2014). Peer instruction in computing: The value of student interaction. *ACM Transactions on Computing Education (TOCE)*, *14*(3), 10:1–10:22.

Zingaro, D., Taylor, C., Porter, L., Clancy, M., Lee, C., Nam Liao, S., & Webb, K. (2018). Identifying student difficulties with basic data structures. In *Proceedings of the 2018 acm conference on international computing education research* (p. 169–177). New York, NY, USA: Association for Computing Machinery. Retrieved from https://doi.org/10.1145/3230977.3231005

Tables

Table 1: Table mapping Inferentialist principles to CSEr.

Context	**Normative rules**	**Inferential entitlements and commitments**	**Deontic score-keeping**
• institution • previous lecture modules; • current lecture modules.	• links between concepts that can be considered tacit knowledge; • links between concepts in the taught content, as it is presented in the teaching and learning environment; • used teaching devices, such as frameworks; • language specific rules: – in the case of programming languages these are stable; – in the case of language determined by the context (such as pseudocode) these are stable in the context; – in the case of natural language these need to judged according to how the student uses language.	• taught content; • premises the student derives from the taught content.	• language dependent: – in the case of natural language it is any claims the student makes, including any inconsistencies; – in the case of pseudocode it is any claims that result in a semantic or syntactic error according to the language of the pseudo-code; – in the case of a programming language it is anything that impairs the execution of the code.

Table 2: Deontic score-keeping for synthetic example.

Entitlement	**Commitment**	**Normative rule violation**	**Implied commitment**
The runtime constraint is $\mathcal{O}(\log_2 n)$.	A binary tree satisfies the requirements.	Runtime is only guaranteed if the tree is self-balancing.	
Tree nodes can hold data; this data can be updated.	Each node holds the access count, which gets updated at every access.		
An inorder traversal can be performed on a tree for insertion of node.	The structure uses in-order traversal to insert a new node.		The structure is not ordered, as it uses in-order traversal for insertion rather than ordering condition.
Removal of a node requires restructuring of the tree.	Child nodes of removed node would need to be promoted; the wording implies this is not implicit to the data structure,		The structure does not restructure itself (by changing references or rebalancing).
An in-order traversal can be used to search through a tree.	The retrieval of the node is done by doing an in-order traversal in the access count.	An in-order traversal by data count on an unordered tree cannot be done in the required runtime.	The structure is not ordered by access count, so all nodes need to be processed until a match is found.
The data count can be updated.	The data count is updated.		The updating of the data count
The runtime constraint is $\mathcal{O}(\log_2 n)$; the tree is not self-balancing.	The runtime is $\mathcal{O}(\log_2 n)$ because of the maximum depth of tree.	The worst case runtime for a depth first traversal of an unbalanced tree is $\mathcal{O}(n)$.	The tree has a depth that does not grow faster than $\log_2 n$.
The runtime constraint is $\mathcal{O}(\log_2 n)$; the tree is not ordered.	The runtime is $\mathcal{O}(\log_2 n)$ because of the maximum depth of tree.	The worst case runtime for a search on an unordered tree is $\mathcal{O}(n)$.	The tree being ordered does not affect runtime.
The algorithm needs to add, remove and retrieve nodes in $\mathcal{O}(\log_2 n)$ time.	The algorithm does all that is required in the required runtime.		

Figures

Figure 1: Diagrammatic representation of deontic score-keeping for an inference

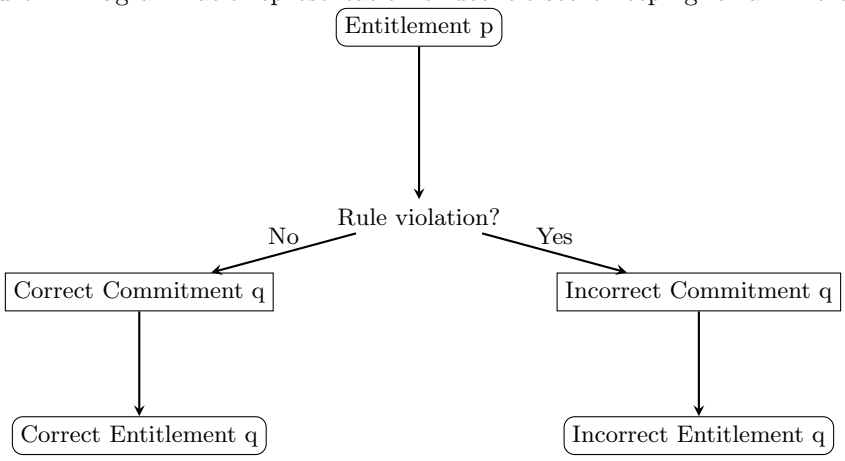

Figure captions

1. Table 1: Table mapping Inferentialist principles to CSEr.
2. Figure 1: Diagrammatic representation of deontic score-keeping for an inference.
3. Table 2: Deontic score-keeping for synthetic example.