Modelling contextuality by probabilistic programs with hypergraph semantics

Peter D. Bruza

School of Information Systems, Queensland University of Technology, GPO Box 2434, Brisbane 4001, Australia

Article Info
Article history:
Received 23 June 2017
Received in revised form 23 October 2017
Accepted 22 November 2017
Available online xxxx

Keywords:
Probabilistic programming
Probabilistic modelling
Programming language semantics
Contextuality

ABSTRACT
Models of a phenomenon are often developed by examining it under different experimental conditions, or measurement contexts. The resultant probabilistic models assume that the underlying random variables, which define a measurable set of outcomes, can be defined independent of the measurement context. The phenomenon is deemed contextual when this assumption fails. Contextuality is an important issue in quantum physics. However, there has been growing speculation that it manifests outside the quantum realm with human cognition being a particularly prominent area of investigation. This article contributes the foundations of a probabilistic programming language that allows convenient exploration of contextuality in wide range of applications relevant to cognitive science and artificial intelligence. Using the style of syntax employed by the probabilistic programming language WebPPL, specific syntax is proposed to allow the specification of “measurement contexts”. Each such context delivers a partial model of the phenomenon based on the associated experimental condition described by the measurement context. An important construct in the syntax determines if and how these partial models can be consistently combined into a single model of the phenomenon. The associated semantics are based on hypergraphs in two ways. Firstly, if the schema of random variables of the partial models is acyclic, a hypergraph approach from relational database theory is used to compute a join tree from which the partial models can be combined to form a single joint probability distribution. Secondly, if the schema is cyclic, measurement contexts are mapped to a hypergraph where edges correspond to sets of events denoting outcomes in measurement contexts. Recent theoretical results from the field of quantum physics show that contextuality can be equated with the possibility of constructing a probabilistic model on the resulting hypergraph. The use of hypergraphs opens the door for a theoretically succinct and efficient computational semantics sensitive to modelling both contextual and non-contextual phenomena. In addition, the hypergraph semantics allow measurement contexts to be combined in various ways. This aspect is exploited to allow the modular specification of experimental designs involving both signalling and no signalling between components of the design. An example is provided as to how the hypergraph semantics may be applied to investigate contextuality in an information fusion setting. Finally, the overarching aim of this article is to raise awareness of contextuality beyond quantum physics and to contribute formal methods to detect its presence by means of probabilistic programming language semantics.

© 2017 Elsevier B.V. All rights reserved.

E-mail address: p.bruza@qut.edu.au.

https://doi.org/10.1016/j.tcs.2017.11.028
0304-3975/© 2017 Elsevier B.V. All rights reserved.

Please cite this article in press as: P.D. Bruza, Modelling contextuality by probabilistic programs with hypergraph semantics, Theoret. Comput. Sci. (2018),
https://doi.org/10.1016/j.tcs.2017.11.028
1. Introduction

Probabilistic models are used in a broad swathe of disciplines ranging from the social and behavioural sciences, biology, the physical and computational sciences, to name but a few. At their very core, probabilistic models are defined in terms of random variables, which range over a set of outcomes that are subject to chance. For example, a random variable \( R \) might be defined to model the performance of human memory. In this case, the possible outcomes might be words studied by a human subject before their memory is cued. After cueing, the subject recalls the first word that comes to mind from the set of study words. This outcome is recorded as a measurement. Repeated measurements over a set of subjects allow the probability of the recall of a certain word to be empirically established.

It is important to note from the outset that the random variable \( R \) has been devised by the modeller with a specific functional identity in mind, namely to model the recall of a set of predefined study words. When developing probabilistic models in this way, the underlying assumption is that the functional identity of a random variable is independent of the context in which it is measured. For example, the purpose, or functional identity of \( R \) is assumed to be the same regardless of whether the memories of human subjects are studied in a controlled laboratory, or in “the wild”, such as in a night club. This assumption seems perfectly reasonable. However, in quantum physics the analog of this assumption does not always hold and has become known as “contextuality”. More formally, the Kochen–Specker theorem [29] implies that quantum theory is incompatible with the assumption that measurement outcomes are determined by physical properties that are independent of the measurement context. Placing this theorem in the context of probabilistic models: contextuality is the “impossibility of assigning a single random variable to represent the outcomes of the same measurement procedure in different measurement conditions” [4].

Contextuality plays a central role in the rapidly developing field of quantum information in delineating how quantum resources can transcend the bounds of classical information processing [28]. It also has important consequences for our understanding of the very nature of physical reality. It is still an open question, however, if contextuality manifests outside the quantum realm. Some authors in the emerging field of quantum cognition have investigated whether contextuality manifests in cognitive information processing, for example, human conceptual processing [21,6,7,12,25] and perception [9,8,37].

It is curious that the preceding deliberations around random variables have a parallel in the field of computer programming languages. More than five decades ago, programming languages such as FORTRAN featured variables that were global. (In early versions of FORTRAN, all variables were global.) As programming languages developed, global variables were seen as a potential cause of errors. For example, in a large program a variable \( X \) can inadvertently be used for functionally different purposes at different points in the program. The error can be fixed by splitting variable \( X \) into two global variables \( X_1 \) and \( X_2 \). In this way \( X_1 \) can be used for one functional purpose and \( X_2 \) for the other, and hence there is no danger that their unique functional identities can become confused. However, when the program involves large numbers of global variables, keeping track of the functional identities of variables can become tedious and a source of error. Such errors were considered serious and prevalent enough that following in the wake of Dijkstra’s famous paper titled “Go To statement considered harmful”, Wulf and Shaw [36] advocated in a similarly influential article that global variables are “harmful” and perhaps should be abolished. This stance was developed in relation to block structured programming languages. A “block”, or “scope”, refers to the set of program constructs, such as variable definitions, that are only valid within a delineated syntactic fragment of program code. Wulf and Shaw [36] argued that when a program employs a scope in which variable \( X \) is defined locally, as well as a variable with the same label \( X \) that is global to that scope, then \( X \) becomes “vulnerable” for erroneous overloading. The theory of programming languages subsequently developed means so that a variable with the same label can be used in two different scopes but preserve a unique functional identity within the given scope. This is not the case in state-of-the-art probabilistic modelling. We believe that the way probabilistic models are currently developed is somewhat akin to writing FORTRAN programs from a few decades ago. For this we mean that in the development of a probabilistic model all the random variables are global. As a consequence errors can appear in the associated model should the functional identity of variables be changing because the phenomenon being modelled is contextual.

The aim of this article to contribute the foundations of a probabilistic programming language that allows convenient exploration of contextuality in wide range of applications relevant to cognitive science and artificial intelligence. For example, dedicated syntax is illustrated which shows how a measurement context can be specified as a syntactic scope in a probabilistic program. In addition, random variables can be declared local to a scope to allow overloading, which is convenient for the development of models. Such programs are referred to a P-programs and fall within the emerging area of probabilistic programming [24].

Probabilistic programming languages (PPLs) unify techniques from conventional programming such as modularity, imperative or functional specification, as well as the representation and use of uncertain knowledge. A variety of PPLs have been proposed (see Gordon et al. [24] for references), which have attracted interest from artificial intelligence, programming languages, cognitive science, and the natural language processing communities [22]. However, unlike conventional programming languages, which are written with the intention to be executed, a core purpose of a probabilistic program is to specify a model in the form of a probability distribution. In short, PPLs are high-level and universal languages for expressing probabilistic models. As a consequence, these languages should not be confused with probabilistic algorithms, or randomized algorithms, which employ a degree of randomness as part of their logic.

In addition to the dedicated syntax, P-programs have a semantics based on hypergraphs which determine whether the phenomenon is contextual. These semantics will be based on hypergraphs in two ways: Firstly, a hypergraph approach from
دریافت فوری متن کامل مقاله

امکان دانلود نسخه تمام متن مقالات انگلیسی
امکان دانلود نسخه ترجمه شده مقالات
پذیرش سفارش ترجمه تخصصی
امکان جستجو در آرشیو جامعی از صدها موضوع و هزاران مقاله
امکان دانلود رایگان ۲ صفحه اول هر مقاله
امکان پرداخت اینترنتی با کلیه کارت های عضو شتاب
دانلود فوری مقاله پس از پرداخت آنلاین
پشتیبانی کامل خرید با بهره مندی از سیستم هوشمند رهگیری سفارشات