Infinite Choice and Probability Distributions An Open Problem: The Real Hotel

Jan Friso Groote
Department of Mathematics and Computer Science
Eindhoven University of Technology
Eindhoven, Netherlands
J.F.Groote@tue.nl

We sketch a process algebra with data and probability distributions. This allows to combine two very powerful abstraction mechanisms namely non-deterministic choice and probabilities. However, it is not clear how to define an appropriate semantics for the generalised choice over data in combination with probability density functions. The real hotel is a puzzle that exemplifies the core of the problem.

1 Process algebra with nondeterministic choice and data

Robin Milner can be seen as the founder of process modelling formalisms, now known as process calculi or process algebras. Well known process algebras are CCS (Calculus of Communicating Systems, [15]), CSP (Communicating Sequential Processes, [10]) and ACP (Algebra of Communicating Processes [2]). We work in the setting of mCRL2, which is a process algebra based on ACP, extended with data, time and probabilities [6].

In a contribution in 1973 Robin Milner [13, 14] pointed out that the primary concept to model systems with behaviour is the *atomic action*, typically denoted as a, b, c, *on*, *off*. Actions are atomic in the sense that they take place at some instant in time. Using process operators, such as sequential composition ('$\cdot$') and alternative composition ('$+$'), actions are combined into behaviour. Recursive behaviour is specified using guarded recursive equations of the form $X = p$ where p is a process expression. For instance the process equation $X = a\cdot(b+c)\cdot X$ has as solution for X the process that can repeatedly do an action a, followed by either an action b or c. It is common to depict such behaviour as a transition system, see Figure 1. There is a special process, denoted δ, (or $\mathbf{0}$ or *NIL* in other process algebras), which represents the deadlocked or inactive process that cannot do anything. By using the parallel operator ('$\|$') processes can be put in parallel, but we do not consider the parallel operator here.

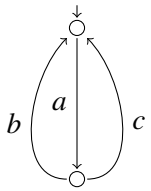

Figure 1: The transition system for X where $X{=}a\cdot(b{+}c)\cdot X$

Milner stressed the importance of *nondeterminism* as an abstract means to model complex behaviour in a simple way. A state is nondeterministic if there are multiple outgoing transitions with the same label to different processes. An example is $a\cdot b + a\cdot c$. Whenever the action a happens, it is unclear whether the action b or the action c can be done next. In the real system that we are describing it may be very clear when which action a is chosen. For instance, the first a is chosen when the temperature is above 20°, and otherwise the second a is performed. However, it might be that we do not want to model the physics behind temperatures, and simply abstract away from it. Sometimes people refer to this nondeterminism by 'Milner's weather conditions'.

Probabilities provide another abstraction technique and they are very different from nondeterminism. When writing $X = (a\cdot b + a\cdot c)\cdot X$, the process X repeatedly performs the nondeterministic a. There is

V. Castiglioni and O. Dardha (Eds.): Combined Workshop on Expressiveness in Concurrency and Structural Operational Semantics (EXPRESS/SOS 2021)
EPTCS 339, 2021, pp. 2–8, doi:10.4204/EPTCS.339.2

some unknown mechanism that determines which a is chosen. It could be that they are chosen with a certain probability, but it is also possible that in every odd round the left a is chosen, and in every even round the right a is selected. Nondeterminism is a more liberal way of modelling than probabilities. We come back on probabilities later.

When modelling realistic systems, it is necessary to be able to use data. Here, the four ways data is added to processes in mCRL2 are explained. Data is specified using applicative equational datatypes [6]. The details are not relevant for us at the moment, although the relationship between abstract datatypes and measurable functions requires further clarification. For the moment it is enough to know that all relevant data types such as natural numbers, integers, reals, lists, messages, functions, etc. can be specified easily.

The first two ways of extending processes with data is by using data in actions and in process equations. Actions can be extended with data as arguments, as in $a(3,[])$, which is the action a with value 3 and an empty list. Actions with arguments behave as ordinary atomic actions, and the only difference is in essence the extended name. Process variables can be parameterised with one or more parameters, for instance as in $Counter(n{:}\mathbb{N}) = up{\cdot}Counter(n{+}1)$, which models a simple counter. Strictly spoken, the variable $Counter$ is a higher order variable ranging over functions from natural numbers to processes.

If process variables have parameters, behaviour should be controllable by the provided data. For this the if-then-else construct is used, which we denote as $c{\to}p{\diamond}q$. The idea is that if condition c is true, the behaviour of p is executed, and otherwise the behaviour of q.

Receiving data from some external party is modelled by writing all possible actions representing the receipt of some data. Concretely, reading either true or false can be denoted by

$$read(true){\cdot}X(true) + read(false){\cdot}X(false)$$

where $X(b{:}\mathbb{B})$ represents some process that expresses what will be done with the received boolean. The outside world indicates whether $read(true)$ or $read(false)$ will take place, indicating which boolean is the input.

This can be generalised to data types with an infinite number of elements, such as $\mathbb{N}$ or $\mathbb{R}$, for which we introduce the generalised sum operator $\sum_{d:D} p$. The process expression

$$\sum_{r:\mathbb{R}} read(r){\cdot}Y(r)$$

represents the uncountably infinite sum over real numbers. External processes can indicate which $read(r')$ must take place, indicating that this r' is the input. Note that the generalised sum operator is a fundamental operator and not an abbreviation. It acts as a binder for the variable r.

The combination of the generalised sum and the if-then-else construct is very expressive, both theoretically [12] and practically. It is for instance easily possible to express that a number smaller than 100 must be read and processed:

$$\sum_{n:\mathbb{N}} (n{<}100) \to read(n){\cdot}Process(n) \diamond \delta.$$

Although the language presented above is concise, it is more than adequate to model realistic behaviour. It forms the essence of the mCRL2 process specification language. This language has been used to model and study many protocols and distributed algorithms. It is for instance also in use as the verification backend of design tools for high quality control software [4, 3].

2 Process algebra with probability distributions

An interesting question is how to extend this process language with the possibility to express probabilities and reason about them. There are a number of languages available where processes have been combined with probabilities, such as Modest [9] and Prism [11]. We follow the approach as outlined in [5]. This approach is partly implemented in the mCRL2 toolset, where state space exploration and reduction for finite probability distributions is possible.

The essential extension is to write

$$\frac{f(d)}{d{:}D}X(d)$$

expressing that a value from a domain D is chosen with probability distribution f and this value is assigned to variable d. This is a concise but flexible notation both allowing for discrete and continuous distributions, not restrained to for instance only exponential distributions as is not uncommon in other probabilistic process algebras. The distribution f has the property that the sum of $f(d)$ over all elements $d{:}D$ equals 1. The notation in the mCRL2 toolset for this construct is **dist** $d{:}D[f(d)].X(d)$.

It is pretty straightforward to describe probabilistic behaviour. We give a number of simple examples. The process throwing a head or tail repeatedly and with equal probability can be characterised by the equation:

$$Throw = (\frac{\frac{1}{2}}{b{:}\mathbb{B}}\ b{\rightarrow}head \diamond tail){\cdot}Throw. \tag{1}$$

One may wonder whether this process is behaviourally equivalent to a process where it is a priori determined that head will be thrown $k \in \mathbb{N}$ times, before tail shows up, where the probability of a sequence of k heads is $\frac{1}{2}^{k+1}$. This latter behaviour is described by the following two process equations:

$$\begin{aligned} &ThrowSequence = \tfrac{\frac{1}{2}^{k+1}}{k{:}\mathbb{N}}Heads(k),\\ &Heads(k{:}\mathbb{N}) = (k>0){\rightarrow}head{\cdot}Heads(k-1) \diamond tail{\cdot}ThrowSequence. \end{aligned}$$

An example taken from [5] shows how timed behaviour can be expressed.

$$install{\lhd}st{\cdot}\frac{\mathcal{N}_0^{\infty}(\mu,\sigma^2)}{t{:}\mathbb{R}}break\_down{\lhd}(st+t){\cdot}\frac{\mathcal{N}_0^{\infty}(\mu,\sigma^2)}{t{:}\mathbb{R}}repair{\lhd}(st{+}t{+}u)$$

where t and u are distributed according to the normal distribution $\mathcal{N}_0^{\infty}(\mu,\sigma^2)$ truncated to the interval $[0,\infty)$ and $a{\lhd}t$ expresses that action a happens at exactly time t. In the mCRL2 tool this is expressed as $a@t$.

This formalism is also suitable for more complex probabilistic processes. For instance in [8] a precise behavioural description is provided of a game of chance called 'The game of the Goose', including an analysis of probabilities of the various geese to win the game. In [7] an intriguing puzzle called the 'Lost boarding pass' is modelled. Tools can easily provide the requested probability that the last passenger that enters the plane will sit on his own seat, even for planes with hundreds of thousands of passengers.

3 Semantics of probabilistic processes based on measurable sets

In [5] the semantics of probabilistic processes has been described but the generalised sum operator has been omitted. In order to understand the semantics of probabilistic processes, we first explain a common approach to probabilistic processes with distributions over finite domains.

The essential idea is that transitions go from states to distributions. These distributions indicate the probability to end up in a particular state after doing the transition. Consider the behaviour of the process *Throw* defined in process equation (1). It is depicted in Figure 2. The behaviour of *Throw* is a distribution σ where with probability $\frac{1}{2}$ the behaviour of the left state, corresponding to doing the action *head*, and with the same probability the behaviour belonging to the right state is selected. Via the actions *head* or *tail* the system ends up in the initial distribution σ, determining which action will take place next. Note that if the number of states is finite, it is fine if σ is a function that for each state provides the probability that that particular state is taken.

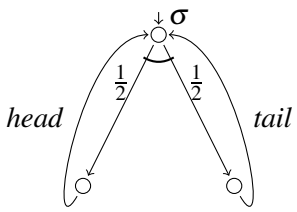

Figure 2: The transition system representing a fair coin

Now consider the following probabilistic process.

$$a\cdot\frac{\lambda e^{-\lambda r}}{r:\mathbb{R}}\ b(r).$$

First, a single a is done, after which the process ends in an exponential distribution determining the distribution of values assigned to the variable r, which is an argument of the subsequent action b. Now the probability of ending up in a state where for some concrete $r'\in\mathbb{R}$ the action $b(r')$ can be done is of course 0. As this is not particularly useful, we provide a density function which provides for sets of states the probability that one of these states can be reached. For our concrete example the probability of ending up in a state where an action $b(r'')$ can be done with $r_1 < r'' < r_2$ is $\int_{r_1}^{r_2}\lambda e^{-\lambda r}dr$.

Abstractly, the distribution σ is a measurable function in a sigma algebra. Measurable functions have a number of pleasant properties, one of them being that the product of two measurable functions is again a measurable function. This allows us to give semantics to most process operators, in particular the choice operator. Consider a process $a\cdot(p+q)$. Processes p and q induce measurable functions σ_p and σ_q that tell the probabilities of the various behaviours that can be done by p and q. The distribution of the process $p+q$ is the function $\lambda X{\subseteq}S.\sigma_p(X)\sigma_q(X)$ where S is the set of all states and X ranges over measurable sets of states.

In [5] a semantics for full mCRL2 with distributions is given, including a notion of strong bisimulation, which is shown to be a congruence. The only operator that was omitted is the generalised sum operator $\sum_{d:D}p(d)$, with D an infinite, not necessarily countable domain. The natural way to define a distribution for this process is to formulate it as the infinite product of all distributions, aka measurable functions belonging to the processes $p(d)$. Concretely, if σ_d is a measurable function indicating the probability density function for process $p(d)$, the natural definition for the probability density function for $\sum_{d:D}p(d)$ would be

$$\lambda X{\subseteq}S.\prod_{d:D}\sigma_d(S).$$

It is however unclear what such a product is, or even whether, or under which circumstances, it actually exists. But in order to work safely with process languages with infinite choice and probability density functions, we require an answer.

4 The real hotel

In order to make the problem of the previous section accessible, we reformulated its essence as a simply looking puzzle formulated as the *real hotel*. The puzzle is an analogy of Hilbert's hotel. Hilbert's hotel

has a countably infinite number of rooms, numbered from 0 upwards. Although the hotel is full, it can still accommodate any additional countable number of persons, for instance, by moving each guest from room n to room $2n$ freeing up all odd numbered rooms.

In the real hotel there are an uncountable number of rooms, numbered with a real number ranging from 0 up to and including 1. All rooms are occupied by the participants of a very successful conference. In the evening they all arrive at their rooms in the hotel where they find a box from which they uniformly draw a real number between 0 and 1. If the drawn number matches the room number, the participant presses a big red button, marked a. If one of the participants presses this button a big letter $\mathcal{A}$ on the roof of the hotel is lit. The question is to determine the probability that the light on the roof of the hotel will switch on.

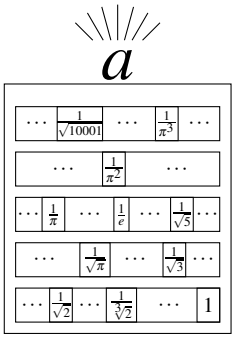

Figure 3: The real hotel

As a process this can be modelled as follows, where for brevity we use $\mathbb{R}_{01}$ as the set of real numbers in the interval $(0,1]$ and $U_{01}(s)$ represents the uniform distribution in the interval $(0,1]$.

$$\sum_{r:\mathbb{R}_{01}} \frac{U_{01}(s)}{s:\mathbb{R}_{01}} (r=s) \to a \diamond \delta. \tag{2}$$

In words, the participant who resides in room r draws a number s and if s equals r the light on the hotel will switch on, represented by the action a. Of course, the probability that a particular guest presses this button is 0. But as this experiment is repeated uncountably often, it is conceivable that the probability that the light on the roof switches on is larger than 0.

Actually, I believe that the behaviour in (2) is exactly bisimilar to the following simpler process, where the value of p is $1-\frac{1}{e} \approx 0.63$:

$$\frac{if(b,p,1-p)}{b:\mathbb{B}}\, b \to a \diamond \delta.$$

We can get confidence that $p = 1-1/e$ by finitely approximating the hotel. Divide the guests in n groups occupying rooms with numbers $(\frac{i}{n}, \frac{i+1}{n}]$ for $i = 0,\ldots,n-1$. The probability that a guest in group i draws a number in group i is $1/n$. The probability that the light stays off is equal to the probability that no participant draws a number in his own group. So, the probability that the light goes on is given by the following formula, where we take $n \to \infty$:

$$\lim_{n\to\infty} 1 - \prod_{i=0}^{n-1} (1-\frac{1}{n}) = 1-\frac{1}{e}.$$

The question is whether this calculation is indeed appropriate. Asking mathematicians with a background in probability theory invariably led to the answer that the puzzle of the real hotel cannot be answered due to the fact that it is not well posed. The real challenge is to set up a theory in which the puzzle of the real hotel can be solved convincingly, or more in general to set up a theory in which process algebra expressions as in (2) have a well defined and understood semantical meaning.

As pointed out in [5], the essential step is to define a density function $[\![p]\!]$which maps the stochastic variables that occur in p to their probability of occurrence. For a process $p+q$, this typically is the product of the density functions for p multiplied by the one for q, which is well defined. For the generalised sum operator $\sum_{d:D} p(d)$ this is the potentially unbounded product of the density functions for $p(d)$. Only a few papers could be found that provide leads to an understanding of such unbounded products [1, 16].

Ignoring the crucial aspect of being well-defined, it is still intriguing to write down examples of probabilistic processes to get an impression of the behaviour that one can describe and want to study.

For instance a slot machine has software initialising its random generator. The random generator $D_t(x)$ determines the payout x but it is influenced by the time t. The process becomes:

$$SlotMachine = \sum_{t:\mathbb{R}^+} wait(t) \cdot \frac{D_t(x)}{x:\mathbb{R}} payout(x) \cdot SlotMachine.$$

An interesting question is to determine the maximal amount of profit that can be obtained by optimal waiting.

There are other conceivable combinations of choice and probability. In a store there are k employees for some fixed number k. The management, not knowing the employees' personalities, uniformly selects one of the employees. Each employee has its own nondeterministic view on how effective he should work with an average output between of l_1 to l_2 outputs per hour. We can describe this by

$$\frac{\frac{1}{k}}{p:Person} \sum_{r:\mathbb{R}_{01}} \frac{(l_1 + r l_2) e^{-l_1 - r l_2 s}}{s:\mathbb{R}} production(p,s).$$

Although we have a clear feeling that this expression is natural, the core question is how to make sure that it has a well defined meaning.

5 Conclusion

We outlined a process algebra with a generalised sum operator and probability distributions. We argued that the semantics of the combination of these two very important modelling notions is not properly understood at the moment. This lack of understanding hampers progress in the field of behaviour with continuous distributions. When the semantics is properly understood, we may open up a whole new field with probabilistic behavioural equivalences, axioms, rules of calculations and verification tools. But most importantly, we will be able to describe and analyse systems with uncountable nondeterminism and probabilities.

Acknowledgements. I like to thank Jan Lanik, Pedro D'Argenio and Pedro Sánchez Terraf for the useful discussions we had regarding this topic.

References

[1] R.J. Aumann (1961): *Borel Structures for function Spaces*. *Illinois Journal of Mathematics* 5(4), pp. 614–630, doi:10.1215/ijm/1255631584.

[2] J.C.M. Baeten & W.P. Weijland (1990): *Process algebra*. *Cambridge tracts in theoretical computer science* 18, Cambridge University Press, doi:10.1017/CBO9780511624193.

[3] R. van Beusekom, B. de Jonge, P. Hoogendijk & J. Nieuwenhuizen (2021): *Dezyne: Paving the Way to Practical Formal Software Engineering*. *EPTCS* 338, pp. 19–30, doi:10.4204/EPTCS.338.4. Proceedings of the 6th Workshop on Formal Integrated Development, F-IDE2021.

[4] Guy H. Broadfoot (2005): *ASD Case Notes: Costs and Benefits of Applying Formal Methods to Industrial Control Software*. In John S. Fitzgerald, Ian J. Hayes & Andrzej Tarlecki, editors: *FM 2005: Formal Methods, International Symposium of Formal Methods Europe, Newcastle, UK, July 18-22, 2005, Proceedings*, *Lecture Notes in Computer Science* 3582, Springer, pp. 548–551, doi:10.1007/11526841_39.

[5] J.F. Groote & J. Lanik (2011): *Semantics, bisimulation and congruence results for a general stochastic process operator*. Computer Science Report 11/05, Department of Mathematics and Computer Science, Eindhoven University of Technology. Available at `https://pure.tue.nl/ws/files/3214338/712142.pdf`.

[6] J.F. Groote & M.R. Mousavi (2014): *Modeling and Analysis of Communicating Systems*. MIT Press, doi:10.7551/mitpress/9946.001.0001.

[7] J.F. Groote & E.P. de Vink (2017): *Problem Solving Using Process Algebra Considered Insightful*. In J.-P. Katoen, R. Langerak & A. Rensink, editors: *ModelEd, TestEd, TrustEd - Essays Dedicated to Ed Brinksma on the Occasion of His 60th Birthday*, *Lecture Notes in Computer Science* 10500, Springer, pp. 48–63, doi:10.1007/978-3-319-68270-9_3.

[8] J.F. Groote, F. Wiedijk & H. Zantema (2016): *A Probabilistic Analysis of the Game of the Goose*. *SIAM Rev.* 58(1), pp. 143–155, doi:10.1137/140983781.

[9] A. Hartmanns & H. Hermanns (2014): *The Modest Toolset: An Integrated Environment for Quantitative Modelling and Verification*. In E. Ábrahám & K. Havelund, editors: *Tools and Algorithms for the Construction and Analysis of Systems - 20th International Conference, TACAS 2014, Held as Part of the European Joint Conferences on Theory and Practice of Software, ETAPS 2014, Grenoble, France, April 5-13, 2014. Proceedings*, *Lecture Notes in Computer Science* 8413, Springer, pp. 593–598, doi:10.1007/978-3-642-54862-8_51.

[10] C.A.R. Hoare (1985): *Communicating Sequential Processes*. Prentice-Hall.

[11] M.Z. Kwiatkowska, G. Norman & D. Parker (2011): *PRISM 4.0: Verification of Probabilistic Real-Time Systems*. In G. Gopalakrishnan & S. Qadeer, editors: *Computer Aided Verification - 23rd International Conference, CAV 2011, Snowbird, UT, USA, July 14-20, 2011. Proceedings*, *Lecture Notes in Computer Science* 6806, Springer, pp. 585–591, doi:10.1007/978-3-642-22110-1_47.

[12] S.P. Luttik (2003): *On the expressiveness of choice quantification*. *Ann. Pure Appl. Log.* 121(1), pp. 39–87, doi:10.1016/S0168-0072(02)00082-9.

[13] R. Milner (1973): *An approach to the semantics of parallel programs*. In: *Convegno di Informatica Teorica, Pisa*, pp. 283–302.

[14] R. Milner (1973): *A mathematical model of computing agents*. In H.E. Rose & J.C. Shepherdson, editors: *Proceedings Logic Colloquium*, North-Holland, pp. 283–302.

[15] R. Milner (1980): *A Calculus of Communicating Systems*. *Lecture Notes in Computer Science* 92, Springer, doi:10.1007/3-540-10235-3.

[16] W. Ott & J.A. Yorke (2005): *Prevalence*. *Bulletin of the American Mathematical Society* 42(3), pp. 263–290, doi:10.1090/S0273-0979-05-01060-8.