INTERFACE

rsif.royalsocietypublishing.org

Research

Cite this article: Maisto D, Donnarumma F, Pezzulo G. 2015 *Divide et impera*: subgoaling reduces the complexity of probabilistic inference and problem solving. *J. R. Soc. Interface* **12**: 20141335.
http://dx.doi.org/10.1098/rsif.2014.1335

Received: 5 December 2014
Accepted: 13 January 2015

Subject Areas:
computational biology

Keywords:
subgoals, hierarchies, model-based reinforcement learning, planning-as-inference, active inference, problem solving

Author for correspondence:
Giovanni Pezzulo
e-mail: giovanni.pezzulo@istc.cnr.it

Divide et impera: subgoaling reduces the complexity of probabilistic inference and problem solving

Domenico Maisto[1], Francesco Donnarumma[2] and Giovanni Pezzulo[2]

[1]Institute for High Performance Computing and Networking, National Research Council, Via Pietro Castellino, 111, 80131 Naples, Italy
[2]Institute of Cognitive Sciences and Technologies, National Research Council, Via S. Martino della Battaglia, 44, 00185 Rome, Italy

GP, 0000-0001-6813-8282

It has long been recognized that humans (and possibly other animals) usually break problems down into smaller and more manageable problems using subgoals. Despite a general consensus that subgoaling helps problem solving, it is still unclear what the mechanisms guiding online subgoal selection are during the solution of novel problems for which predefined solutions are not available. Under which conditions does subgoaling lead to optimal behaviour? When is subgoaling better than solving a problem from start to finish? Which is the best number and sequence of subgoals to solve a given problem? How are these subgoals selected during online inference? Here, we present a computational account of subgoaling in problem solving. Following Occam's razor, we propose that good subgoals are those that permit planning solutions and controlling behaviour using less information resources, thus yielding parsimony in inference and control. We implement this principle using approximate probabilistic inference: subgoals are selected using a sampling method that considers the descriptive complexity of the resulting sub-problems. We validate the proposed method using a standard reinforcement learning benchmark (four-rooms scenario) and show that the proposed method requires less inferential steps and permits selecting more compact control programs compared to an equivalent procedure without subgoaling. Furthermore, we show that the proposed method offers a mechanistic explanation of the neuronal dynamics found in the prefrontal cortex of monkeys that solve planning problems. Our computational framework provides a novel integrative perspective on subgoaling and its adaptive advantages for planning, control and learning, such as for example lowering cognitive effort and working memory load.

1. Introduction

The human problem-solving literature shows that an often-used strategy to solve complex problems is *subgoaling*: splitting the original problem into smaller and more manageable tasks whose achievement corresponds to 'milestones' or 'subgoals' of the original problem [1]. This strategy is ubiquitous while solving problems of different kinds. For example, navigational planning problems can be decomposed by using landmarks (e.g. known locations) as subgoals and puzzles such as the Tower of Hanoi can be decomposed by using subgoals such as 'free up third rod'.

In cognitive neuroscience, there is a general consensus that subgoals are key to complex planning and problem solving. Most theories of executive function assume that goals and subgoals are maintained in prefrontal hierarchies and permit exerting cognitive control during demanding activities [2–5]. It has been reported that the monkey prefrontal cortex encodes a sequence of activation of subgoals and goals during a delay period prior to action [6,7]. Failure to find good task decomposition or to follow the right sequence of subgoals can produce poor planning performance, as frequently observed in prefrontal patients solving the Tower of Hanoi [8].

Computational studies also provide many demonstrations of the usefulness of subgoals. In the hierarchical reinforcement learning (HRL) literature, it has been consistently shown that, given the right decomposition, problems can be learned and solved more efficiently, that is, in less time and with less resources [9,10]. The most popular HRL framework, called the 'Options' framework, explicitly uses subgoals for building *temporal abstractions* that allow faster learning and planning [11]. An Option can be conceptualized as a sort of macro-action that includes a list of starting conditions, a policy and a termination condition (subgoal). It is well assessed in the HRL literature that Options and subgoals bring several advantages in terms of faster learning and a reuse of strategies across problems (or skill transfer) [12]. However, this is only true if the right subgoals are selected: the selection of the wrong subgoals can be deleterious and slow down learning and inference [13].

Subgoals are not only useful for learning but they might also facilitate cognitive control. A recent series of studies used information-theoretic notions to measure the amount of information in working memory that (sub)goals can 'save' while at the same time allowing successful task completion. In this formulation, the most useful subgoals are those that allow controlling behaviour with less information in memory; from a cognitive perspective, these advantages might be linked to lower working memory requirements during monitoring and control [14].

Finally, subgoaling is essential for planning and problem solving. Most popular architectures for symbolic problem solving [1,15] and planning [16,17] include a subgoaling component; and subgoaling is also used in a few connectionist systems [18,19]. It is often assumed (especially in the human problem solving/AI tradition [1,15]) that subgoaling proceeds backward from the final goal states and serves to resolve 'impasses': if a goal-achieving action cannot be executed because of a missing precondition, achieving the precondition becomes the next system subgoal (and so on). Less investigated is the case of the feed-forward (i.e. from the current to the goal state) selection of subgoals during planning, which is recognized as an important determinant of human problem solving [20,21] and whose neuronal underpinnings have been studied in monkeys [6,7] and humans [22].

We offer a model of subgoal selection that starts from a normative principle: we propose that during planning subgoals should be selected that minimize the computational complexity [19,23] of the problem at hand. In our proposed approach, subgoaling proceeds in a feed-forward way with the objective of maximizing the probability of the resulting sub-problems (or *programs*, see later) by selecting those ones with minimum algorithmic complexity and description length. Below we describe the computational approach and successively we validate it in two computational experiments, which focus on two issues: (i) under which conditions using subgoals leads to optimal behaviour and (ii) whether the specific subgoaling mechanisms envisaged here can be linked to neuronal computations in primates.

2. Material and methods

Our approach to subgoaling is framed within the *planning as probabilistic inference* framework, where goal-directed planning and problem solving are cast in terms of probabilistic inference [24–26]. In this framework, planning is usually implemented by imposing ('clamping') future goal states as observations, and running a probabilistic inference until a solution is found that goes from the current to the goal state [27]. In other words, planning consists of selecting actions that bias the probability of future events towards desirable states. A variant of this method consists of clamping future rewards rather than goal states [28,29]. Here, we use a related method that stems from Active Inference theory [30]: we set goals as (high) Bayesian *priors* and not as future observations, with the advantage that one has more flexibility on *when* (i.e. after how many steps) the goal will be eventually achieved.

In the remainder of this section, we introduce the three key components of the proposed method: (i) the probabilistic model used for the inference, which describes the conditional dependencies between the relevant stochastic variables (e.g. states, actions and subgoals); (ii) the method we adopted to assign 'prior' values to the subgoal variable; and (iii) the probabilistic inference method.

2.1. Probabilistic model

The probabilistic model we adopt is the Dynamic Bayesian Network (DBN [31]) of figure 1. In this representation, the nodes are stochastic variables expressing the elements that influence the planning process. They are

— S describes the agent's *states*; for example, its position in a maze. Here, we assume a discrete set of states, represented as integer values in the range $\{0, \ldots, n\}$, with n being the total number of states.

— SG represents the *subgoals* (or *subgoal states*) used for planning. Potentially, every state S can be a subgoal used to achieve the final goal; this implies that SG can assume the same values as S: $\{0, \ldots, n\}$. We assume that the final goal state is a particular subgoal having the highest *a priori* probability (see below).

— A represents the *actions* that an agent can execute to move from a state to another state. In the two-dimensional mazes we use for our simulations, the agent has five actions: $\{u, d, l, r, \varepsilon\}$. The first four actions represent moving up, down, left and right, respectively. ε is a 'rest' action, or a transition from a state to the same state.

— Π represents the *policies* (or mappings between states and actions) available to the agent [32]. Intuitively, policies permit specification of sequences of transitions from (say) a start and a goal state. The number m of policies available to the agent vary depending on the number of states and actions in the environment. We include also a *rest policy* π_ε associated with each state the action ε.

— F describes the agent's progress to the goal state, i.e. whether it has *finished* or not. The allowed values of F are $\{0, 1, 2\}$. The value is 2 if the agent has reached the final goal state (and is an *absorbing* state preventing further inference); 1 if the agent has just reached a subgoal state; 0 otherwise. The main role of this variable is monitoring subgoal achievement and guiding the transitions between subgoals [33].

Overall, the model corresponds to a two-layered DBN (where the nodes are arranged on two layers corresponding to two consecutive slices of time indicated with subscripts, e.g. S_t and S_{t-1}). First-order and stationary Markov properties hold: the conditional probabilities do not change as a function of t and every variable depends exclusively on other variables expressed at the same or the immediately preceding time step. In our simulations, the DBN structure and parameters are assumed to be known. Note that the transition $P(\pi|\mathrm{SG}, S)$ captures the concept of an Option

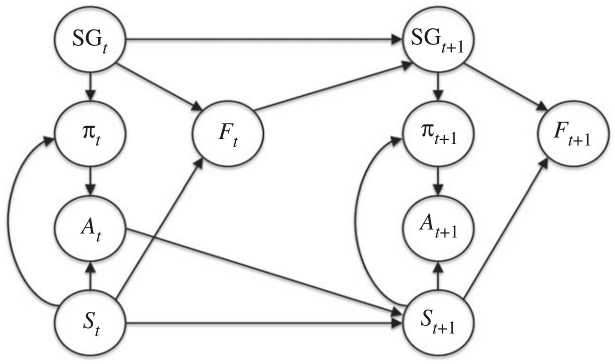

Figure 1. Graphical model (DBN [31]) used for the simulations. See the main text for explanation.

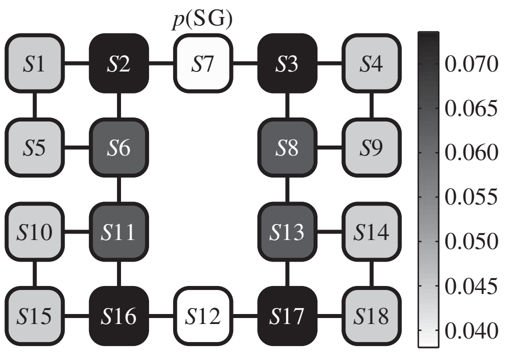

Figure 2. The four-rooms scenario used in Experiment 1. It includes 18 states $S = \{S1, \ldots, S18\}$ and is composed of four 'rooms' with a single connection among them ($S7$ and $S12$). Subgoal priors used in Experiment 1 are shown in grey scales; $S18$ is the goal state. The probability values are: 0.03 for $S7$, $S12$, 0.04 for $S1$, $S4$, $S5$, $S9$, $S10$, $S14$, $S15$, $S18$, 0.06 for $S6$, $S8$, $S11$, $S13$ and 0.08 for $S2$, $S3$, $S16$, $S17$.

in HRL, but is expressed in a probabilistic way; furthermore, here the Options are not 'cached' but formed on-the-fly.

2.2. How to generate algorithmic priors for subgoals

A key departure from current models using the *planning-as-inference* framework is that we address the problem of subgoal selection. Here, subgoals are states that are inferred during the inference and guide the selection of compact and effective sequences of transitions from the initial state to the subgoal(s) and ultimately to the final goal. By assuming that the state space is discrete, such transitions can be considered as series of *programs* necessary to execute a series of moves from the agent's initial state s_0 to the first subgoal state sg_1, to the second sg_2, and so on (following the instructions contained in some *policy* π_j) until the agent's final goal state s_{goal} is achieved.

It is important to note that every program can formally be transformed into a *code*, a binary string p of length $|\mathrm{p}|$ that can be processed by a computing machine. Thus, following principles of algorithmic probability theory [23,34] (see appendix A for a discussion), one can associate a probability $2^{-|\mathrm{p}|}$ to every binary program p, depending on the program length. By taking into account every program returning the subgoal sg, built with every possible combination between input states and policies, the algorithmic probability for a subgoal sg can be computed, up to a normalization factor, by

$$P(\mathrm{sg}) \propto \sum_i \sum_j 2^{-|\mathrm{p}(s_i,\pi_j)|}. \tag{2.1}$$

The above-equation does not concern just subgoal states; it can be applied to every state belonging to S, in order to generate an *a priori* 'algorithmic probability' distribution (hereafter also denoted as *algorithmic priors*) that assigns a prior to each state, with highest priors given to the most strongly informative ones, i.e. the subgoals, important for decomposing any multistep planning task. This approach is conceptually similar to the method described in [35] that considers for each state 'the amount of Shannon information that the agent needs to maintain about the current goal at a given state to select the appropriate action'.

Algorithmic priors for subgoal states depend on how many programs halt in those states and how long they are, and are thus specific for a given environment. The computations required to compute algorithmic priors for subgoals in a specific environment are very costly, but there are ways of alleviating this problem. First, algorithmic priors can be computed with a significantly reduced computational cost (see appendix B) in an off-line way (in batch), once and for all, and can be successively re-used for every inference in the same environment. Second, there are effective approximations to the exact calculations. One can uniformly sample a set of policies, to be used in the algorithmic probability formula, with a cardinality of some order less than π and (given a sufficient number of samples) this procedure approximates well the prior distributions calculated by enumerating every potential policy. The fidelity of the priors depend on the computational resources used to calculate them and can be improved trial by trial making new observations, as evident in the 'cumulative' nature of equation (2.1) and of equation (A 1) in appendix A.

Finding subgoals that might be '*a priori*' good is only the first step of our method. A key distinction in the proposed approach is between *potentially useful subgoals* and *selected subgoals*, the former being *a priori* and independent of the agent's current goal and the latter dependent on it. To understand this difference, consider the 18-states 'four-rooms' planning problem shown in figure 2, which is often used as a testbed in HRL scenarios [11]. Most studies have assessed that specific states such as *bottlenecks* (in this case, the 'doors' $S2$, $S3$, $S16$ and $S17$) are often useful subgoals and permit splitting the search space down into more manageable subproblems [12,13,35]. Our proposed method using equation (2.1) yields similar results, as represented by the colour gradient of figure 2 (the darker the colour, the higher the probability). In this perspective, although all the states can potentially be subgoals, some of them (e.g. bottlenecks or doors) are more likely candidates and have higher probability.

However, knowing the potentially useful subgoals is not sufficient to solve a specific problem, because the best subgoals for each specific planning problem depend on the start and goal states and not only on *a priori* information on the environment. Consider, for example, an agent solving the problem of planning how to go from $S15$ to $S18$ in the 'four-rooms' scenario of figure 2. Potentially, all 'doors' are useful subgoals but given the specific start and goal states the best choice would be passing through $S16$ and $S17$, not $S2$ or $S3$. The inferential procedure should thus preferentially find a plan that traverses these *selected subgoals* and not the others. We describe next the inference permitting to do so.

Algorithm 1: Planning inference (s_0, s_{goal}, p(SG), T_{max})

```
Require: Agent's initial state $s_0$, goal state $s_{goal}$, subgoal algorithmic priors $p$(SG), maximum number of forward inferences $T_{max}$.
Ensure: State sequence $[s_{0,\ldots,}s_{goal}]$, subgoal sequence Seq.
    $t = 0$
    set $S_0$ to the agent's initial state $s_0$
    sample a subgoal $sg_0$ from the prior distribution $p$(SG) attained by using equation (2.1) on each state
    select a policy $\pi_0$ maximizing equation (2.2) sampled through a Monte Carlo method
```

```
determine the action $a_0$ depending on $\pi_0$ and $s_0$
evaluate the termination condition state $F_0$ according to
    $p(F_0|sg_0, s_0)$
while ($F_t \neq 2$ and $t \leq T_{\max}$) do
    $t = t + 1$
    determine the state $s_t$ by means of $p(S_t|a_{(t-1)}, s_{(t-1)})$
    select the subgoal $sg_t$ maximizing equation (2.3) sampled
        through a Monte Carlo method
    select a policy $\pi_t$ maximizing equation (2.2) sampled
        through a Monte Carlo method
    determine the action $a_t$ depending on $\pi_t$ and $s_t$
    evaluate the termination condition variable $F_t$ according
        to $p(F_t|sg_t, s_t)$
end while
```

2.3. Inferential procedure

The aim of the proposed *planning as probabilistic inference* scheme is inferring a plan from the agent's initial state s_0 to its final goal state s_{goal}, which are assumed to be known (say, from $S1$ to $S18$ in the 'four-rooms' scenario). Normally, in HRL this would correspond to finding a suitable policy from $S1$ to $S18$. Rather, here we consider the possibility that the plan actually consists of a series of smaller plans, say, one from $S1$ to $S16$ and one from $S16$ to $S18$. In this case, the inference would result in a first subgoal ($S16$) along with a suitable policy permitting the transition from $S1$ to $S16$; and a second subgoal ($S18$), which actually corresponds to the final goal, along with a suitable policy permitting the transition from $S16$ to $S18$. Importantly, here we use considerations of algorithmic probability to select subgoals and policies; as a consequence, the plan results in series of transitions across states with highest algorithmic probability.

The inference uses the probabilistic model of figure 1 to iteratively *sample* (i.e. extract probabilistically) candidate subgoals from the previously described *a priori* subgoal distribution (see §2.2); the candidate subgoals are then retained or discarded by considering the computational complexity of the resulting sub-problems or *programs*. When a new subgoal is selected, it becomes the starting point for a new iteration, until the final goal is eventually found. The iterative sampling procedure runs, for a maximum number of steps $T_{\max}$, through all the problem space and returns a sequence of subgoals and associated programs that essentially solve the original problem by finding sequences of smaller sub-problems (but in principle, a sample might also generate a solution without subgoals, from the start to the goal state).

The pseudocode of Algorithm 1 describes the procedure step-by-step. Essentially, using the model of figure 1, the inference uses the agent's initial state s_0 as a clamped (i.e. observed) state on the node S at the time $t = 0$, and the agent's goal state s_{goal} as the state with highest prior on the subgoal node SG.

Initially, at the time $t = 0$, a subgoal sg_0 is drawn over the *a priori* algorithmic probability distribution described in §2.2. Then, for each time t, the inference proceeds by finding out a policy π_t so that a program permitting a transition from s_t to sg_t can be built. Formally, this corresponds to drawing a policy from the probability distribution $p(\Pi_t|s_t, \text{sg}_t)$. Using Bayesian calculus and assuming that there is no prior information about policies (see appendix A, equation (A 2)), $p(\Pi_t|s_t, \text{sg}_t)$ can be approximated by the following expression:

$$\begin{aligned} p(\Pi_t = \pi_j|s_t, \text{sg}_t) &\propto p(\text{sg}_t, s_t|\Pi_t = \pi_j)p(\pi_j) \\ &= p(\text{sg}_t|s_t, \pi_j)p(s_t|\pi_j)p(\pi_j). \end{aligned} \tag{2.2}$$

According to the above equation, the probability of selecting a specific policy π_j depends (in addition to its prior probability) on the length of the *program* generated with π_j that from the current state s_t takes us to the currently examined subgoal sg_t, weighted by the s_t prior computed just over π_j. Drawing from the distribution defined in equation (2.2) would require to reckon every distribution value and this would be very costly and often infeasible for even moderately large state spaces. To overcome this problem, we use a Monte Carlo sampling method [36] to collect a set of policies accepted as realizations of equation (2.2) by using the uniform distribution $p(\Pi_t)$ as a proposal distribution (note that the sampling method can be conducted using a parallel procedure). From this set we select the policy $\pi^* = \text{argmax}_j\, p(\pi_j|s_t, \text{sg}_t)$. Note, however, that alternative methods such as heuristic techniques or tree search might be adopted to this purpose [37].

To sum up, the inference described so far starts from an initial (clamped) state and returns a subgoal and a policy to reach it. In other words, it builds an Option-like plan on-the-fly that would (ideally) correspond to the shortest possible program from the initial to the subgoal state. Once this program is established, one can get the transition to the time $t + 1$ off the ground. The action a_t is directly determined for passing from the state s_t to the next one s_{t+1}.

Simultaneously, the node F_t monitors (sub)goal achievement and guides the transitions $\text{SG}_t \rightarrow \text{SG}_{t+1}$ between subgoals [33]. If $s_t \neq \text{sg}_t$, the value of F_t is zero and the subgoal sg_{t+1} is forced to be the same as of the time t. When the current state s_t is equal to the selected subgoal sg_t, the *rest policy* π_ε (i.e. a specific policy associating with every state a 'rest' action ε) is selected determining $s_{t+1} = s_t = \text{sg}_t$.

Then, a new subgoal (SG_{t+1}) has to be chosen. To guide subgoaling towards the final goal state (rather than, say, away from it), not only the inference 'clamps' the just-achieved subgoal as the current state, but it also assumes that $f_{t+1} = 2$, that is, it fictively assumes that the goal state is (has to be) observed. The procedure then continues as explained earlier, until the final goal state s_{goal} is reached (and $f_t = 2$).

All these operations are summarized by the following distribution:

$$p(\text{SG}_{t+1}|f_t, \text{sg}_t) = \begin{cases} \delta_{\text{sg}_k\, \text{sg}_t} & \text{if } f_t = 0 \\ p(\text{SG}_{t+1} \mid f_{t+1} = 2, \text{sg}_t) & \text{if } f_t = 1 \\ \delta_{\text{sg}_k\, s_{\text{goal}}} & \text{if } f_t = 2, \end{cases} \tag{2.3}$$

where δ_{ab} is 1 when $a = b$ while is 0 otherwise, and $p(\text{SG}_{t+1}|f_{t+1} = 2, \text{sg}_t)$ is estimated by a Monte Carlo process [36] applied to the posterior probability distribution of SG_{t+1}:

$$p(\text{SG}_{t+1} = \text{sg}_k \mid f_{t+1} = 2, \text{sg}_t) \propto p(f_{t+1} = 2 \mid \text{sg}_k)\, p(\text{sg}_k \mid \text{sg}_t) \tag{2.4}$$

considering that $p(\text{sg}_k|\text{sg}_t)$ estimates the probability that sg_k succeeds sg_t as subgoal and that the evaluation of the likelihood $p(f_{t+1} = 2|\text{sg}_k)$ corresponds to computing the conditional probability $p(g|\text{sg}_k)$ of the goal state with respect to the subgoal sg_{t+1} (for a more detailed description, see appendix A, especially equation (A 3), and appendix B). Here, the value $\text{sg}^* = \text{argmax}_k\, p(\text{sg}_k|f_{t+1} = 2, \text{sg}_t)$, sampled by adopting the algorithmic priors of equation (2.1) as proposal distribution, is set as subgoal at time $t + 1$. Note that using this procedure the probability of selecting an optimal policy increases when the state S_t becomes closer to the subgoal.

3. Results

3.1. Experiment 1: four-rooms scenario

A first question we ask is whether the proposed subgoaling-based method is more efficient compared to an equivalent procedure without subgoaling that searches for a solution to the problem from start to finish, and under which conditions it leads to optimal behaviour (e.g. the selection of shorter paths). To answer this question, we compare the *planning-as-*

Table 1. Parameters used in Experiments 1 and 2.

parameters	Experiment 1	Experiment 2
states	18	21
actions	5	5
policies	$\sim 6.7 \times 10^6$	$\sim 5.4 \times 10^9$
instances	50, 100, 1000	100
T_{max}	12	8
$P(S_{goal})$	0.095	0.052

inference method with and without subgoals in the 18-states 'four-rooms' scenario shown in figure 2. Note that even in this apparently simple scenario the number of potential policies is around seven million, making exact inference impracticable.

In the first (*without-subgoals*) strategy, the probability of choosing a subgoal different from the goal state is zero: $p(\text{SG} \neq s_{\text{goal}}) = 0$. In the second (*with-subgoals*) strategy, the above-defined *algorithmic priors* of the subgoals are used: $p(\text{SG} \neq s_{\text{goal}}) > 0$. To study the generality of the method, we tested three different ensembles (of 50, 100 and 1000 of independent inferential instances) and made 10 simulation runs for each. The parameters used in the simulations are reported in table 1. In the experiment, we assume $S1$ as starting state and $S18$ as goal. We set the prior probability $S18$ to be the highest among all the values of $p(\text{SG})$ (table 1). Specifically, we set $P(S18)$ as the highest value of $p(\text{SG})$ plus a small value calculated as the difference between the first and second highest values of $p(\text{SG})$; then we normalize all the other probabilities in $p(\text{SG})$ to sum up to 1.

To compare the inference with and without subgoals, we consider various factors: the number of successfully reached goals, the optimality of behaviour, expressed here as the number of inferential instances achieving the optimal planning strategy (in this scenario, a sequence of eight states, including both start and goal states), the complexity of the inference (i.e. the percentage of instances failing to arrive at the goal) and the complexity of the control (i.e. the average length of the programs necessary to achieve the task).

The differences between the percentage of successful strategies and the percentage of achieved optimal paths carried out in the *without-subgoals* and *with-subgoals* modalities are shown in table 2. The latter strategy achieves a better performance and the largest number of optimal paths by using subgoals to split the search space. In keeping, the programs produced by the strategy *with-subgoals* (dotted black line) are on average shorter than those produced by the strategy *without-subgoals* (grey line), especially in the first inferential steps (figure 3*a*). Here the results are relative to the programs used to plan (up to the next subgoal) at each inferential step, for $N = 100$ instances and averaged on 10 runs. Results are similar with 50 and 1000 instances (not shown).

The average percentage of instances that fail to find a successful planning strategy (for $N = 100$) is shown in figure 3*b*, revealing again an advantage of the strategy *with-subgoals* (dotted black line) over the strategy *without-subgoals* (grey line), especially in the first inferential steps. Furthermore, the fact that the percentage of failures is stable in all the steps suggests that the strategy *with-subgoals* is robust.

Overall, these results highlight that the proposed method using subgoals is more efficient compared to a standard *planning-as-inference* procedure that does not use subgoals and makes a more parsimonious use of resources (e.g. working memory).

We next analysed the solutions found by the proposed method using subgoals. The distribution of the number of subgoals found by the strategy *with-subgoals* (averaged on the different runs) is shown in figure 3*c*. The results show that the strategies including two subgoals before the actual goal (with a total of three sub-plans) are on average the most successful. This result reflects the specificity of the four-rooms scenario; for example, successful cases of navigation between $S1$ (start) and $S18$ (goal) include the subgoal sequences [$S2$, $S3$, $S18$] and [$S16$, $S17$, $S18$].

As a final remark, we note that the probability distributions of S and SG—as estimated by considering the average of 100 instances during the inference—change at every step, as shown in figures 4 and 5, respectively. In the next experiment, we ask whether these dynamic representations might have equivalents in the brain of primates executing planning tasks.

3.2. Experiment 2: neuronal correlates of planning in monkey lateral prefrontal cortex

Few neurophysiological studies have directly probed the role of goals and subgoals during planning and control at the single cell level (see [4] for a review). Perhaps the most direct test of the neuronal underpinnings of planning in prefrontal circuits comes from two monkey studies performed by Tanji and collaborators [6,7]. In these studies, monkeys performed a multistep path planning task in a maze (shown in figure 6*a*) and were required to prepare multiple movements of a cursor stepwise from a starting position (at the centre of the maze) to a pre-instructed goal position which varied in the experimental conditions (see labels from G1 to G8 in figure 6*a*). These experiments revealed that during the preparatory period monkey lateral prefrontal cortex (lPFC) neurons encoded sequential representations of the path plans [7] and that these prefrontal representations are specific for goals and not motor actions (i.e. the position to be reached and not the cursor movement to be executed), in contrast with representations in the primary motor cortex that encoded arm movements [6].

Most importantly for our study, the experiment reported in [7] identified two neuronal populations in the lPFC that are crucial for preparation of path planning: one in which neurons specifically coded the position within the maze that represented the final path goal independent of the steps required to achieve it (called *final goal-selective* neural activity) and one in which neurons specifically coded the position within the maze to which the animal intended to move the cursor next, independent of the movement required or the specific maze configuration (called *immediate goal-selective* neural activity). The aim of our modelling study is identifying whether the dynamics of two key variables used in our model, $P(\text{SG})$ and $P(S)$, might correspond to the activity of *final* and *immediate* goal-selective neurons in the monkey study, respectively, thus providing a computationally motivated explanation of what these neuronal populations might encode during the planning task.

We mapped the maze used in the monkey experiment of [7] (figure 6*a*) into a discrete domain that includes 21 states and four actions (giving approx. 10^9 possible policies), and

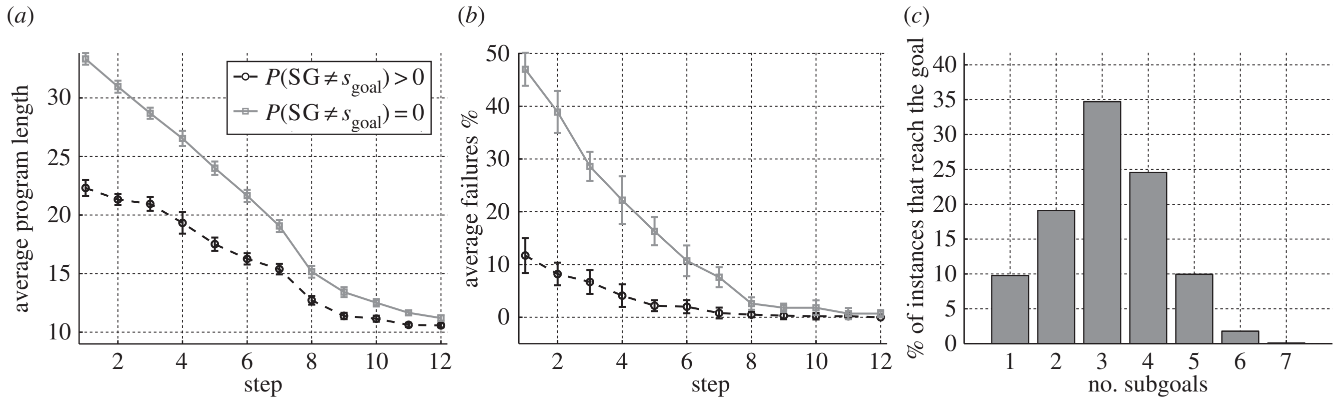

Figure 3. Performance in the four-rooms scenario. (*a*) Average length of programs produced at each inferential step by the two alternative models (without versus with subgoals); standard deviations are shown. The length is coded in terms of binary program length [34]. (*b*) Average failure percentage of instances that do not represent a successful strategy (standard deviation shown). (*c*) Distribution of number of subgoals used by successful strategies. Note that the goal is included so a subgoal of 1 indicates that no additional subgoals were selected.

Table 2. Percentage of instances that correctly find a plan to the goal in Experiment 1. Results are shown for different number of independent inferential instances (50, 100 and 1000).

	% of success with standard deviation without subgoals: $P(SG \neq s_{goal}) = 0$			% of success with standard deviation with subgoals: $P(SG \neq s_{goal}) > 0$		
	step			step		
no. instances	8	10	12	8	10	12
50	34 ± 8	64 ± 6	81 ± 3	50 ± 6	79 ± 6	94 ± 4
100	35 ± 4	63 ± 3	82 ± 3	49 ± 6	80 ± 3	94 ± 2
1000	35 ± 2	65 ± 1	84 ± 1	50 ± 1	81 ± 1	93 ± 1

we calculated the *algorithmic prior* of each state using the aforementioned method; the results are shown in figure 6*b*.

We next conducted a simulated experiment on a sample trial (equivalent to one of those studied in [7]), which consists of running an inference from a starting position ($S11$) to a known goal location ($S3$) (figure 7). The prior probability $S3$ is set to be the highest value (table 1) and all the other probabilities in p(SG) are normalized to sum up to 1. Results are for 10 execution runs using 100 instances with the parameters reported in table 1. Note that this example is chosen for illustrative purposes but the results reported below generalize to all the start-goal pairs of the maze used in the original monkey experiment.

The former panel of figure 7 shows the prior SG once the goal ($S3$) is known; the other panels show how the distribution $p(\mathrm{SG}_t)$ changes during the probabilistic path-planning inference. It is possible to appreciate that during the inference the final goal position $S3$ is tonically maintained in a way that closely mimics the *final goal-selective neural activity* reported by Saito *et al.* [7]. Importantly, while in the first panel the final goal $S3$ is clamped, in the successive steps it is not, but it results from sampling over the $p(\mathrm{SG}_t)$ distribution (e.g. $p(\mathrm{SG}_{t=2}|s_{\mathrm{start}} = S11,\ s_{\mathrm{goal}} = S3)$ in the second inferential step (see §2.3 for details). The fact that the probabilistic inference of SG_t in our proposed algorithm predicts the final goal-selective neural population found in monkeys lPFC [7] suggests that these neurons might be a signature of an ongoing probabilistic inference [38,39] rather than only playing a role in working memory, as usually assumed. This result provides a novel perspective on how prefrontal goal representations might support inference and control, which is in line with other evidence that goal-coding neurons peak before an action is made but are also maintained tonically after action onset [4,40].

The probability distribution of the states $S_{t=1} - S_{t=4}$ in the same path planning example as before (i.e. from $S11$ to $S3$) is shown in figure 8. It is possible to appreciate that these probability distributions closely mimic the *immediate goal-selective neural activity* reported in [7], with a preferential coding of the next spatial position in the maze required to reach the goal. This suggests that the activity of immediate goal-selective neurons codes (predicted) state representations; these prospective representations might complement the activity of final goal-selective neurons in a coherent probabilistic path-planning inference [41]. Importantly, these state representations maintained by lPFC are not just 'reachable' future states but compose optimal (i.e. shorter) plans, in keeping with previous theoretical proposals [42]. Supporting this idea is the fact that the probability distributions found by our method preferentially select optimal plans. For example, in the second step (figure 8*b*) the probability of $S6$ is higher than $S12$. This fact can be easily justified by making simple probabilistic considerations: the probability $P(S3|S7, S6)$ that $S3$ can be reached through the path [$S11$, $S6$, $S7$, $S3$] is greater than the probability $P(S3|S7, S12)$. Indeed, there are two optimal paths passing through $S6$ while there is only one through

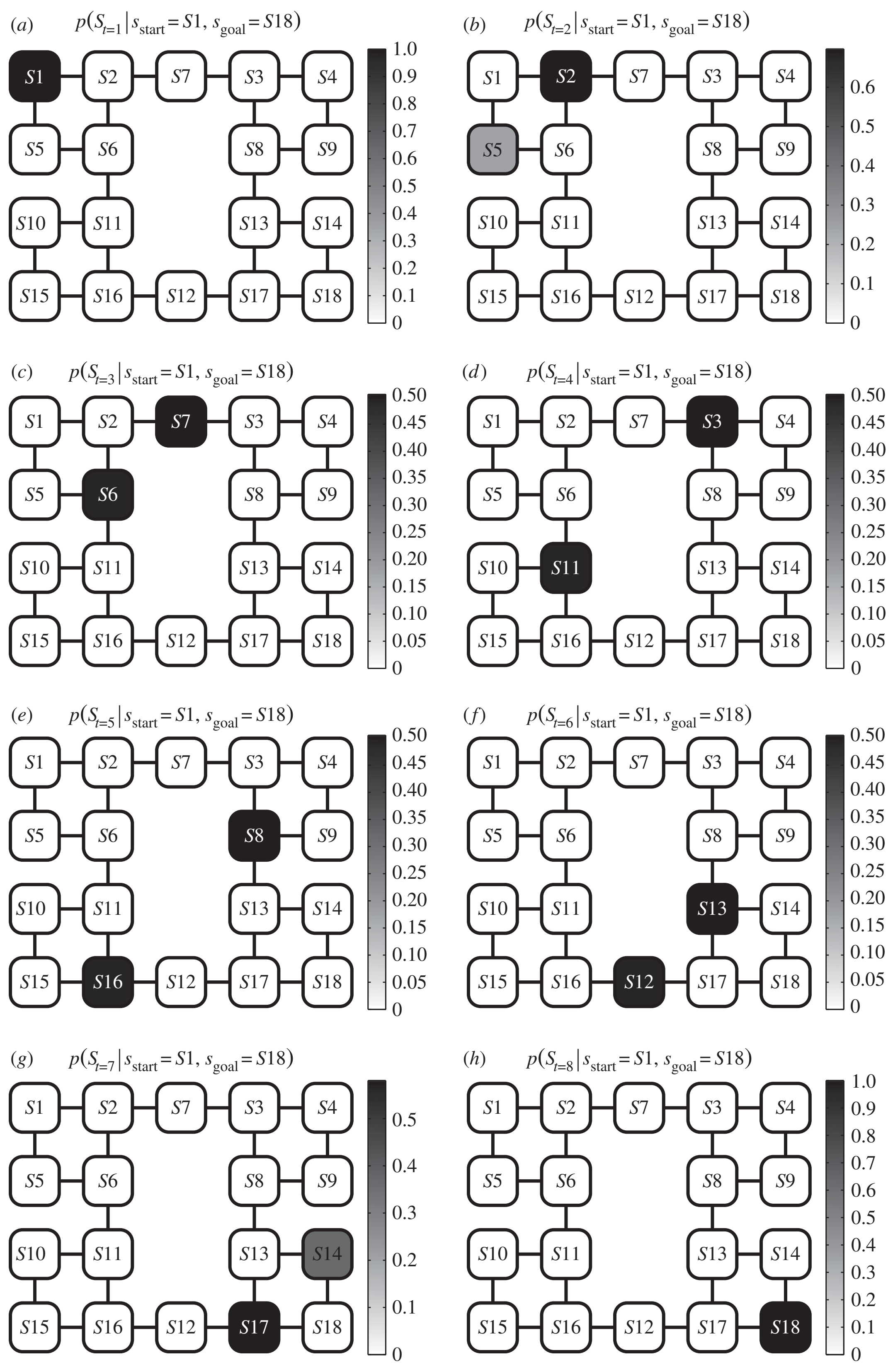

Figure 4. Probability distribution of *S* in the four-rooms scenario, calculated by considering the average of 100 instances at each time step (numbered from *a* to *h*).

*S*12, which would imply the path [*S*11, *S*12, *S*7, *S*3] (and thus visiting *S*6) increases the probability of finding an optimal path to the desired goal.

Overall, this second study shows that the two distributions $p(SG)$ and $p(S)$ used in our probabilistic inference method can be fruitfully linked to two distinctive (final and immediate) goal-selective neural populations found in monkeys lPFC, and provide a novel interpretation of *what* these populations might code during the task (i.e. subgoal distributions and states that compose optimal plans) and *how* these neuronal responses might be produced (i.e. through probabilistic inference). Note that the inference used to compute $p(SG)$ and $p(S)$ is not conditionally dependent on action, which implies that these populations are goal- and not movement-related, as reported in the target monkey study [7].

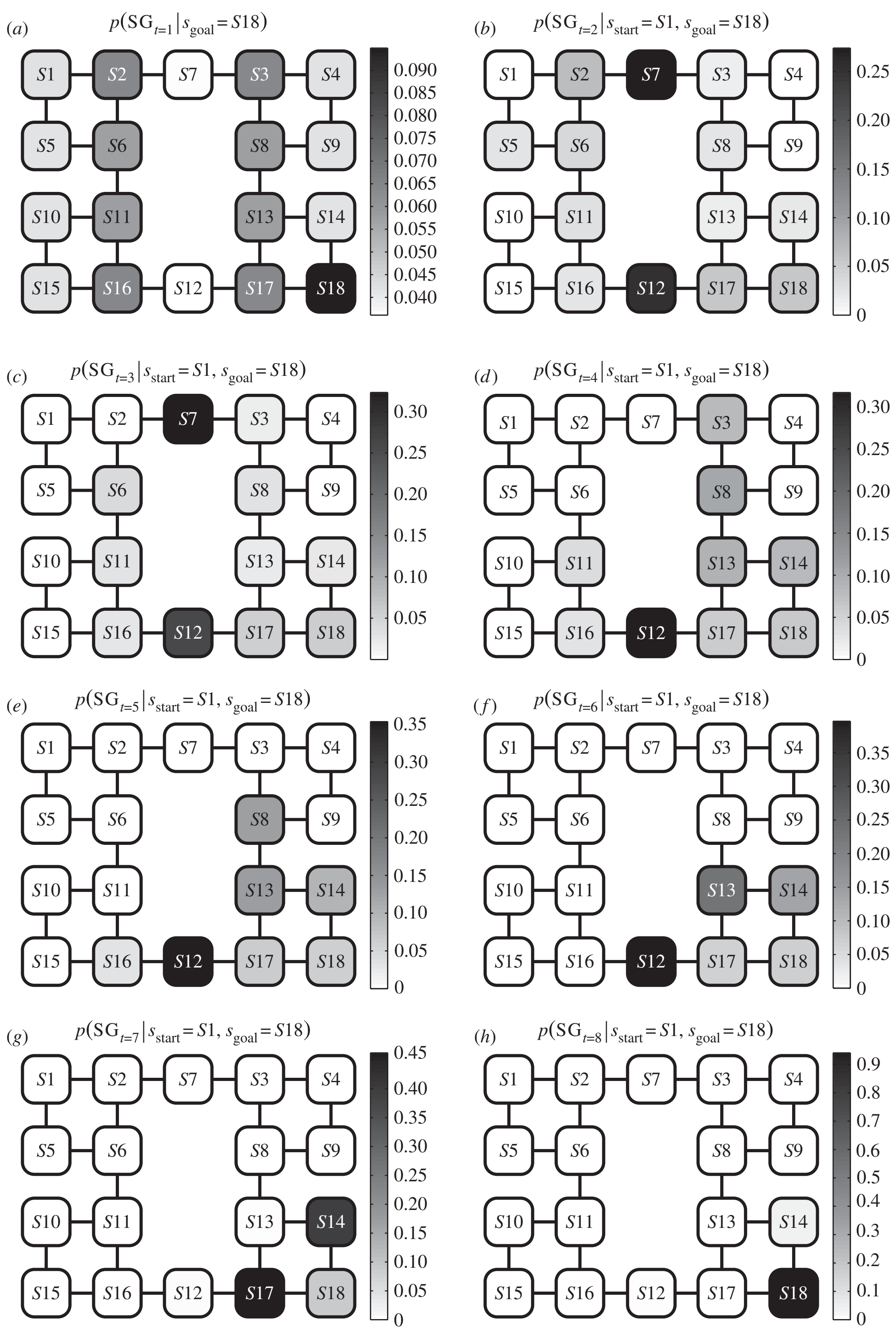

Figure 5. Probability distribution of SG in the four-rooms scenario, calculated by considering the average of 100 instances at each time step (numbered from *a* to *h*).

4. Discussion

The benefits of subgoaling have long been recognized by many disciplines that include psychology, neuroscience and computational modelling, but its computational and neuronal principles are incompletely known. From a normative perspective, we proposed that subgoaling permits planning solutions and control behaviour using less information by selecting more compact *programs*. To implement this idea, we devised a probabilistic inference method that combines *planning as inference* with information-theoretic measures based on Solomonoff's Algorithmic Probability and Kolmogorov Complexity (KC), the latter being widely used in AI [19,23] and closely related to variational probabilistic methods that minimize free energy [30,43]. Essentially, our approach formalizes the 'Occam's razor' principle: *a priori*, among the strings (here, programs) that represent the procedures

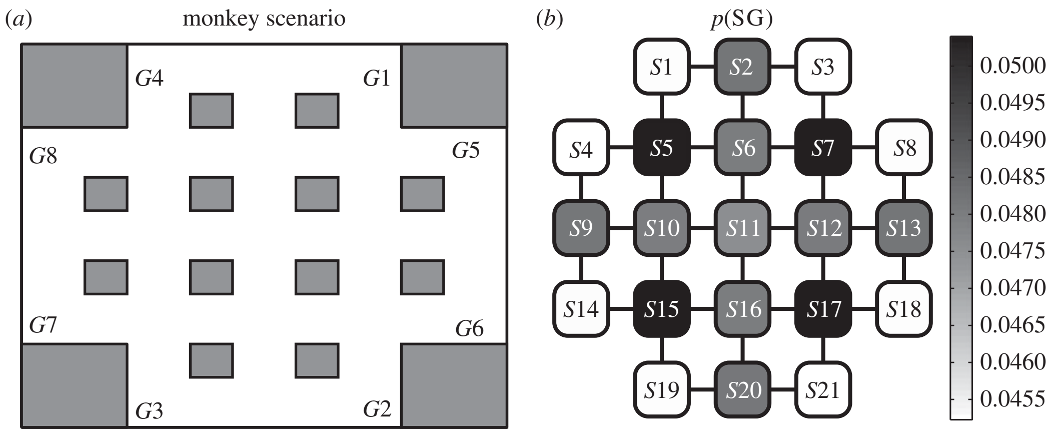

Figure 6. The maze of Experiment 2. (*a*) The original maze used in the monkey study of [7]. Here the start position is the centre and G1–G8 denote possible goal locations. (*b*) The representation of the maze used in Experiment 2. The grey scales indicate the SG *a priori* probability distribution for the task: *S*1, *S*3, *S*4, *S*8, *S*14, *S*18, *S*19, *S*21 has probability 0.045, *S*11 has probability 0.048, *S*6, *S*10, *S*12, *S*16 has probability 0.0485, *S*2, *S*9, *S*13, *S*20 has probability 0.0486 and *S*5, *S*7, *S*15, *S*17 has probability 0.05.

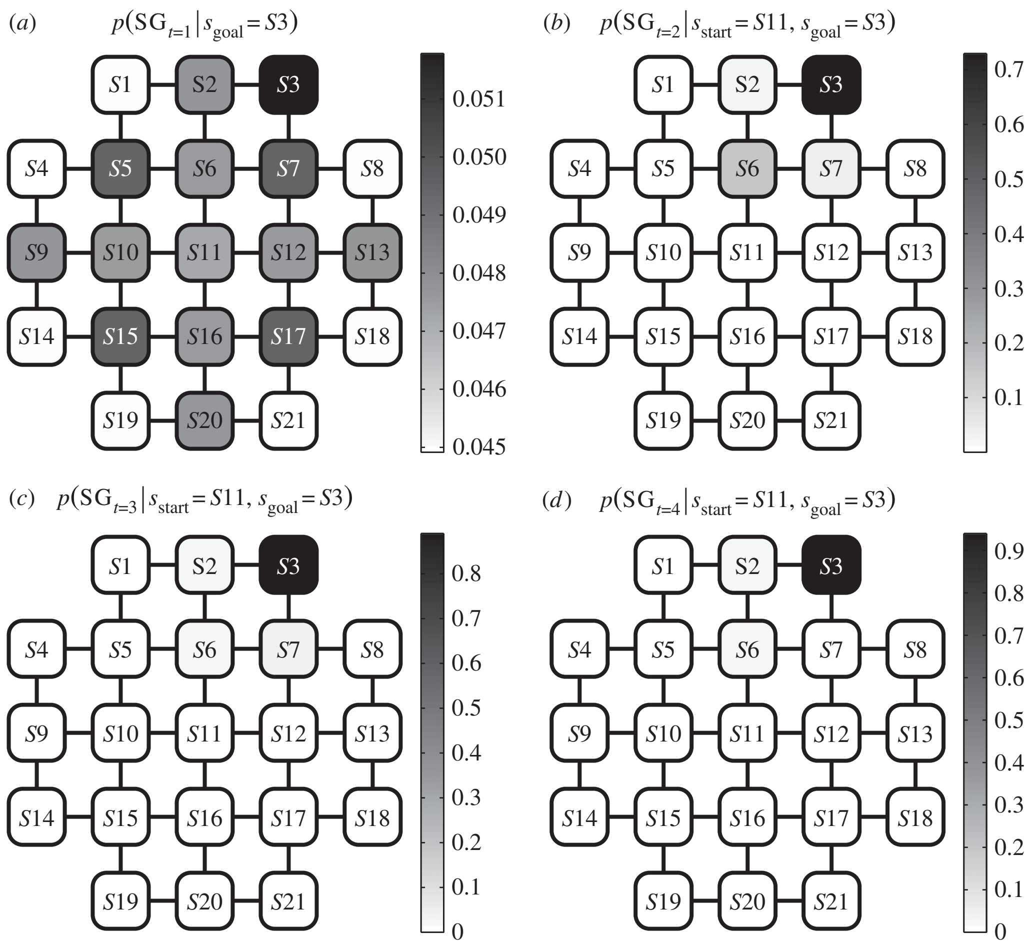

Figure 7. SG probability distribution at each time step (numbered from *a* to *d*) for the monkey path-planning task reported in [7].

returning the same output, 'simpler' strings (i.e. strings with low descriptive complexity) are more probable and should more likely be selected by the probabilistic inference.

There is increased attention in theoretical neuroscience to the idea that brain computations, and in particular state inference and planning, can be understood in terms of probabilistic inference; still, the required computations are computationally expensive and it is unclear how the brain might solve or approximate them [27,30,33,38]. Within this framework, we have demonstrated that parsimony principles derived from normative (information-theoretical) considerations can guide efficient subgoaling and significantly lower the descriptive complexity of inferential procedures underlying planning and problem solving (Experiment 1). We have also linked aspects of the proposed method to planning dynamics found in monkey lPFC (Experiment 2). The proposed computational framework thus offers a mechanistic explanation of subgoaling in planning and problem solving and suggests parallels with neurobiological findings, which remain to be tested in future studies.

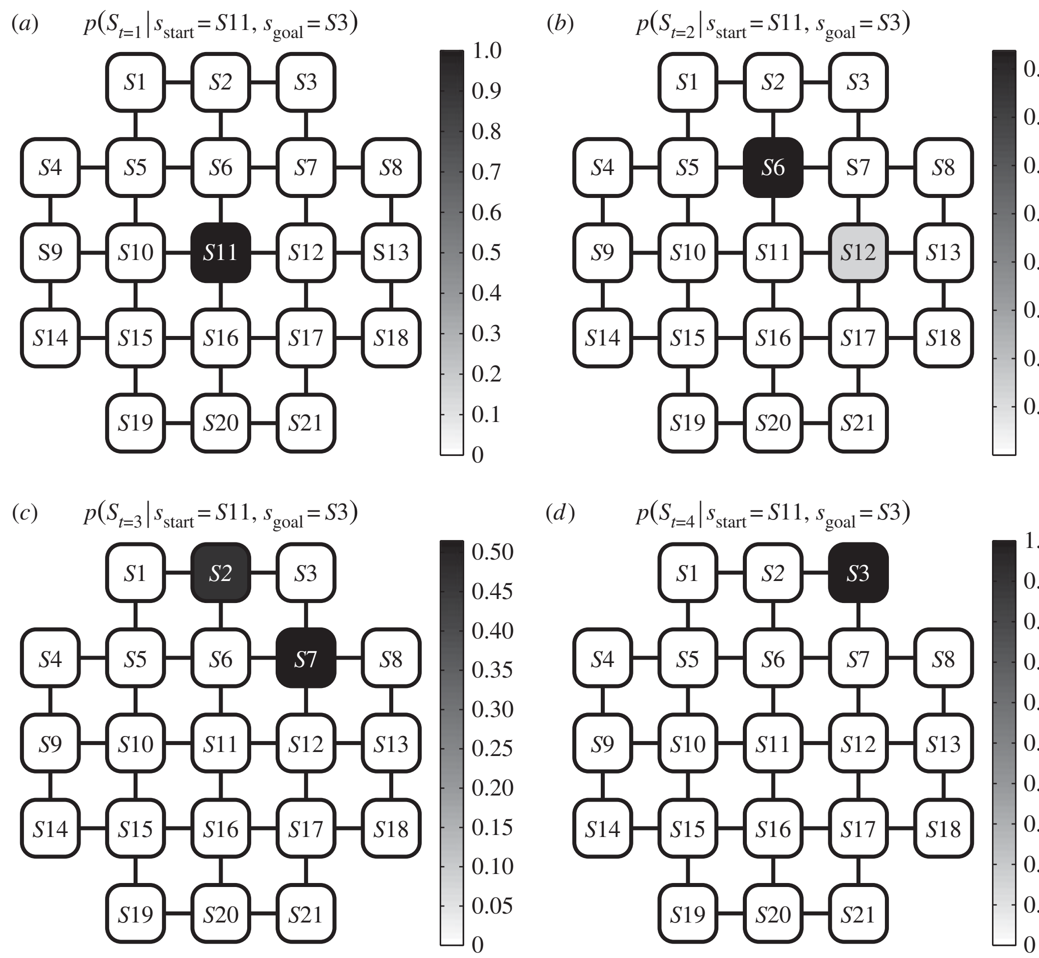

Figure 8. *S* probability distribution at each time step (numbered from *a* to *d*) for the monkey path-planning task reported in [7].

The subgoaling approach proposed here shares some similarities with 'task partitioning' strategies that have been reported in insect societies [44–46]. In insect societies, a task is split into smaller tasks that are carried out in a distributed manner by several individuals (e.g. insects), which can collectively produce optimal solutions. The similarities between the collective decision-making strategies of insect societies and of neurons in the brain have often been noticed [47,48]; understanding the benefits of distributed schemes for subgoaling is an open interrogative for future research.

The proposed method represents a novel approach to planning and subgoaling that can also inform technical fields such as AI/HRL. In HRL, temporal abstraction methods such as Options are widely used but they are normally modelled as structures that, once learned, are retrieved in an all-or-nothing way. On the contrary, our method permits building Options-like constructs *on the fly* using probabilistic inference, which yields increased flexibility. Note that the two approaches are not mutually exclusive but can be combined. The method presented here can be easily extended to model the acquisition of stable or 'cached' skills or Options; it is sufficient to allow the system to store the results of previous inferences (e.g. in the form of *priors* on policies) and re-use them in similar future inferences. Running inferences based on such prior information would generate a well-known dilemma between faster but less flexible (habitual) and slower but more flexible (goal-directed) selection mechanisms [26,49–53].

Although we have focused on the importance of subgoaling for planning, the same computational principles might operate at the three different timescales of planning, control and learning. These three problems are traditionally studied in isolation but are closely related in probabilistic schemes and might use common optimization principles, including the proposed subgoal-based, *Divide et Impera* (divide and conquer) strategy. During planning, subgoals (or Options in HRL) reduce the search space by permitting planning at a higher level of temporal abstraction (i.e. at the level of macro-actions and not only basic actions). During control, subgoals permit maintaining the smallest possible information in working memory that is sufficient for successful task achievement [35]. During learning, subgoals (and associated Options) permit more efficient learning, reducing the search space, which also helps in the transfer of knowledge and skills to novel domains [12].

An open research question is whether goals and subgoals might support *efficient coding* and information compression in cortical hierarchies. The rationale of this idea is that, in the same way perceptual coding might be based on *minimum description length* and efficient coding principles [54], information compression principles might explain how the brain efficiently codes control programs [55]. In this perspective, goals and subgoals might ensure that information is compressed in a way that is more useful to solve problems efficiently and thus bias which control programs (e.g. Options) should be learned. It remains to be tested in future studies whether plans that are initially formed using probabilistic inference and subgoaling might be successively stored in prefrontal hierarchies, yielding an efficient coding of control-related information.

Funding statement. Research funded by the EU's FP7 under grant agreement no FP7-ICT-270108 and the HFSP under grant agreement

RGY0088/2014. The GEFORCE Titan used for this research was donated by the NVIDIA Corporation.

Appendix A. Algorithmic probability, Bayesian inference and subgoaling

As usually in computer science, transition sequences can be considered as series of instructions that let a computing machine return an output for some input, provided that the execution halts. In the spirit of Algorithmic Probability Theory introduced by Solomonoff [23,34], we assume that (in a discrete state space) it is possible to determine a set of executable instructions by considering a starting state s_i, an arrival state $s_{i'}$ and a policy π_j. The policy is represented as a state-action table encoding a list of actions that, by starting from s_i, will get to $s_{i'}$. In algorithmic terms, the triple $(s_i, \pi_j, s_{i'})$ defines a *program* for the state $s_{i'}$. This correspondence is not biunique because the same program can be determined by different policies.

The next step consists of encoding the program in a computable format. A simple and efficient way to do so is using the bitwise encoding. By mapping the program $(s_i, \pi_j, s_{i'})$ into a binary string $\mathrm{p}_{i'}(s_i, \pi_j)$ with length $|\mathrm{p}_{i'}(s_i, \pi_j)|$, we can algorithmically assign to it an *a priori* probability equal to $2^{-|\mathrm{p}_{i'}(s_i,\pi_j)|}$. This probability goes to 0 in the limit of $|\mathrm{p}_{i'}(s_i, \pi_j)| \to \infty$ (e.g. if a program does not halt in $s_{i'}$ or it does not halt at all) and is equal to 1 when $|\mathrm{p}_{i'}(s_i, \pi_j)| = 0$ (i.e. when initial and arrival states coincide). In our representation, this last condition can happen if and only if there exists a policy working as an 'identity function', namely having the transition $s_i \to s_i, \forall i$ (that corresponds to the policy π_ε, in our approach).

By considering the set $\{\mathrm{p}_{i'}(s_i, \pi_j)\}_{i,j}$ of all the programs returning the state $s_{i'}$, the *a priori* algorithmic probability of $s_{i'}$ can be defined as

$$P(s_{i'}) = \sum_i \sum_j \left(\frac{2^{-|\mathrm{p}_{i'}(s_i,\pi_j)|}}{\sum_l 2^{-|\mathrm{p}_l(s_i,\pi_j)|}} \right), \qquad \text{(A 1)}$$

when the above-equation is applied on every state, it generates an *a priori* probability distribution, depending on the specific domain, in which the subgoal states have the highest probability values.

This algorithmic method to define probabilities can be used to determine the conditional probability distribution commonly involved in Bayesian inference. For example, let us consider equation (2.2) of §2.3: we can obtain the expressions for $p(\mathrm{sg}_t|s_i, \pi_{j)}$ by setting s_t and π_j in equation (A 1); and the expression for $p(s_t|\pi_j)$ by setting s_t in equation (A 1). Therefore, equation (2.2) can be rewritten in an algorithmic form as

$$\begin{aligned} p(\Pi_t = \pi_j|s_t, \mathrm{sg}_t) &\propto p(\mathrm{sg}_t|s_t, \pi_j)p(s_t|\pi_j)p(\pi_j) \\ &= \frac{1}{Z}(2^{-|\mathrm{p}_{\mathrm{sg}_t}(s_t,\pi_j)|}) \left(\sum_i 2^{-|\mathrm{p}_{s_t}(s_i,\pi_j)|} \right) p(\pi_j) \\ &= \frac{1}{Z} \left(\sum_i 2^{-(|\mathrm{p}_{\mathrm{sg}_t}(s_t,\pi_j)|+|\mathrm{p}_{s_t}(s_i,\pi_j)|)} \right) p(\pi_j), \end{aligned} \qquad \text{(A 2)}$$

with Z as a normalization constant. According to the above equation, the probability of a specific policy π_j, given s_t and sg_t, is proportional to its aptitude to generate programs of minimum length which, starting from every possible state s_i, permit movement to the currently examined subgoal sg_t by passing through the current state s_t.

Analogously, one can determine the algorithmic expression of $p(\mathrm{SG}_{t+1} = \mathrm{sg}_k|f_{t+1} = 2,\ \mathrm{sg}_t)$ in equation (2.4) by constraining all the programs returning the goal to have sg_k as an intermediate outcome. Equation (2.4) thus becomes

$$\begin{aligned} p(\mathrm{SG}_{t+1} = \mathrm{sg}_k|f_{t+1} = 2, \mathrm{sg}_t) &\propto p(f_{t+1} = 2|\mathrm{sg}_k)p(\mathrm{sg}_k|\mathrm{sg}_t) \\ &\equiv p(g|\mathrm{sg}_k)p(\mathrm{sg}_k|\mathrm{sg}_t) \\ &= \frac{1}{Z'} \left(\sum_j 2^{-|\mathrm{p}_g(\mathrm{sg}_k,\pi_j)|} \right) \left(\sum_j 2^{-|\mathrm{p}_{\mathrm{sg}_k}(\mathrm{sg}_t,\pi_j)|} \right), \end{aligned} \qquad \text{(A 3)}$$

where Z' is a normalization constant.

It is worth noting that equation (A 3) is equivalent, up to the normalization factor, to $p(s_{\mathrm{goal}|}\mathrm{sg}_t)$ when $\mathrm{sg}_k = s_{\mathrm{goal}}$ or $\mathrm{sg}_k = \mathrm{sg}_t$. Besides, it holds 1 when $s_{\mathrm{goal}\,=}\,\mathrm{sg}_t$. This is a direct consequence of the fact that, when initial and final states coincide, the only policy able to determine a halting program of length zero is the 'rest policy' π_ε with probability equal to 1. Rather, when the rest policy is fixed, the probability of a program is different from zero if and only if the starting and final states are the same.

Evaluating equation (A 1) or (A 3) is computationally burdening as it depends on the number m of policies, which is usually prohibitive also for a space with few states. However, it is possible to introduce various approximations and heuristics, such as that presented in appendix B, to render their evaluation computationally treatable.

Appendix B. Heuristic procedure for estimating algorithmic conditional probabilities

Equations (A 1) and (A 3) have a considerable computational cost, because they are evaluated over the whole set of the policies. However, it is possible to use heuristics to decrease the cost of calculating the conditional probabilities used in the two equations.

The state space of the problem can be arranged as a graph having the states s_i as the n vertices and the transitions e between states as the edges. The graph can be represented as an *adjacency list*: a collection of unordered lists, one for each state, composed of the neighbours of the related vertex. Given two states, s_i and $s_{i'}$, we exploit a standard depth-first search algorithm for figuring out all the paths $c \in C(s_i, s_{i'})$ between them. Every path c corresponds intuitively to a distinct program p_c, but each program can be attained by different policies.

As a consequence, various policies give the same contribution to the algorithmic probability computation. Their number can be estimated by considering every possible combination of the neighbours of the states not involved in the specified path c (and, thus, not present in the program p_c). We call this value *multiplicity* of the program p_c and we indicate it as $\mu(\mathrm{p}_c)$.

Therefore, the probability $p(s_{i'}|s_i)$ can be written in the following form:

$$p(s_{i'}|s_i) = \sum_{c \in C(s_{i'},s_i)} \mu(\mathrm{p}_c)(2^{-|\mathrm{p}_c|}), \qquad \text{(B 1)}$$

which makes the conditional probability estimation affordable with a computational complexity of $O(n + e)$, equal to the depth-first search complexity. An analogous procedure can be used to calculate $p(\mathrm{sg}_k|\mathrm{sg}_t)$ and $p(g|\mathrm{sg}_k)$.

References

1. Newell A, Simon HA. 1972 *Human problem solving*. Englewood Cliffs, NJ: Prentice-Hall.
2. Fuster JM. 1997 *The prefrontal cortex: anatomy, physiology, and neuropsychology of the frontal lobe*. Philadelphia, PA: Lippincott-Raven.
3. Miller EK, Cohen JD. 2001 An integrative theory of prefrontal cortex function. *Annu. Rev. Neurosci.* **24**, 167–202. (doi:10.1146/annurev.neuro.24.1.167)
4. Passingham RE, Wise SP. 2012 *The neurobiology of the prefrontal cortex: anatomy, evolution, and the origin of insight*, vol. 50. Oxford, UK: Oxford University Press.
5. Pezzulo G, Castelfranchi C. 2009 Thinking as the control of imagination: a conceptual framework for goal-directed systems. *Psychol. Res.* **73**, 559–577. (doi:10.1007/s00426-009-0237-z)
6. Mushiake H, Saito N, Sakamoto K, Itoyama Y, Tanji J. 2006 Activity in the lateral prefrontal cortex reflects multiple steps of future events in action plans. *Neuron* **50**, 631–641. (doi:10.1016/j.neuron.2006.03.045)
7. Saito N, Mushiake H, Sakamoto K, Itoyama Y, Tanji J. 2005 Representation of immediate and final behavioral goals in the monkey prefrontal cortex during an instructed delay period. *Cereb. Cortex* **15**, 1535–1546. (doi:10.1093/cercor/bhi032)
8. Shallice T. 1982 Specific impairments of planning. *Phil. Trans. R. Soc. Lond. B* **298**, 199–209. (doi:10.1098/rstb.1982.0082)
9. Botvinick MM. 2008 Hierarchical models of behavior and prefrontal function. *Trends Cogn. Sci.* **12**, 201–208. (doi:10.1016/j.tics.2008.02.009)
10. Solway A, Diuk C, Cordova N, Yee D, Barto AG, Niv Y, Botvinick MM. 2014 Optimal behavioral hierarchy. *PLoS Comput. Biol.* **10**, e1003779. (doi:10.1371/journal.pcbi.1003779)
11. Sutton RS, Precup D, Singh S. 1999 Between MDPs and semi-MDPs: a framework for temporal abstraction in reinforcement learning. *Artif. Intell.* **112**, 181–211. (doi:10.1016/S0004-3702(99)00052-1)
12. Barto AG, Mahadevan S. 2003 Recent advances in hierarchical reinforcement learning. *Discr. Event Dyn. Syst.* **13**, 341–379. (doi:10.1023/A:1025696116075)
13. Botvinick M, Niv Y, Barto A. 2009 Hierarchically organized behavior and its neural foundations: a reinforcement learning perspective. *Cognition* **119**, 262–280. (doi:10.1016/j.cognition.2008.08.011)
14. van Dijk SG, Polani D, Nehaniv CL. 2011 Hierarchical behaviours: getting the most bang for your bit. In *Advances in artificial life. Darwin Meets von Neumann*, pp. 342–349. Berlin, Germany: Springer.
15. Rosenbloom PS, Laird JE, Newell A. 1992 *The soar papers: research on integrated intelligence*, vols. 1 and 2. Cambridge, MA: MIT Press.
16. Hauskrecht M, Meuleau N, Kaelbling LP, Dean T, Boutilier C. 1998 Hierarchical solution of Markov decision processes using macro-actions. In *Proc. 14th Conf. on Uncertainty in artificial intelligence, Madison, WI, 24–26 July*, pp. 220–229. San Francisco, CA: Morgan Kaufmann Publishers Inc.
17. Lipovetzky N, Geffner H. 2012 Width and serialization of classical planning problems. In *ECAI 2012: 20th European Conf. on Artificial Intelligence, Montpellier, France, 27–31 August*, pp. 540–545. Clifton, VA: IOS Press. (doi:10.3233/978-1-61499-098-7-540)
18. Donnarumma F, Prevete R, Trautteur G. 2012 Programming in the brain: a neural network theoretical framework. *Connect. Sci.* **24**, 71–90. (doi:10.1080/09540091.2012.684670)
19. Schmidhuber J. 1997 Discovering neural nets with low Kolmogorov complexity and high generalization capability. *Neural Netw.* **10**, 10–15. (doi:10.1016/S0893-6080(96)00127-X)
20. Mumford MD, Schultz RA, Van Doorn JR. 2001 Performance in planning: processes, requirements, and errors. *Rev. Gen. Psychol.* **5**, 213–240. (doi:10.1037/1089-2680.5.3.213)
21. Spitz HH, Minsky SK, Bessellieu CL. 1984 Subgoal length versus full solution length in predicting Tower of Hanoi problem-solving performance. *Bull. Psychon. Soc.* **22**, 301–304. (doi:10.3758/BF03333826)
22. Simon DA, Daw ND. 2011 Neural correlates of forward planning in a spatial decision task in humans. *J. Neurosci.* **31**, 5526–5539. (doi:10.1523/JNEUROSCI.4647-10.2011)
23. Li M, Vitâanyi PMD. 2008 *An introduction to Kolmogorov complexity and its applications*. Berlin, Germany: Springer.
24. Attias H. 2003 Planning by probabilistic inference. In *Proc. 9th Int. Workshop on Artificial Intelligence and Statistics, Key West, Florida, 3–6 January*. New Jersey: Society for Artificial Intelligence and Statistics.
25. Botvinick M, Toussaint M. 2012 Planning as inference. *Trends Cogn. Sci.* **16**, 485–488. (doi:10.1016/j.tics.2012.08.006)
26. Pezzulo G, Rigoli F, Chersi F. 2013 The mixed instrumental controller: using value of information to combine habitual choice and mental simulation. *Front. Psychol.* **4**, 92. (doi:10.3389/fpsyg.2013.00092)
27. Toussaint M, Storkey A. 2006 Probabilistic inference for solving discrete and continuous state Markov decision processes. In *Proc. 23rd Int. Conf. on Machine learning, Pittsburgh, Pennsylvania, 25–29 June*, pp. 945–952. New York, NY: ACM.
28. Botvinick MM, An J. 2008 Goal-directed decision making in prefrontal cortex: a computational framework. In *Advances in Neural Information Processing Systems (NIPS), Vancouver, Canada, 8–11 December*. Cambridge, MA: MIT Press.
29. Solway A, Botvinick MM. 2012 Goal-directed decision making as probabilistic inference: a computational framework and potential neural correlates. *Psychol. Rev.* **119**, 120–154. (doi:10.1037/a0026435)
30. Friston KJ, Daunizeau J, Kiebel SJ. 2009 Reinforcement learning or active inference? *PLoS ONE* **4**, e6421. (doi:10.1371/journal.pone.0006421)
31. Murphy KP. 2002 Dynamic Bayesian networks: representation, inference and learning. PhD thesis, Computer Science Division, UC Berkeley, USA.
32. Sutton RS, Barto AG. 1998 *Reinforcement learning: an introduction*. Cambridge, MA: MIT Press.
33. Verma D, Rao RPN. 2006 Planning and acting in uncertain environments using probabilistic inference. In *IROS, Beijing, China, 9–15 October*, pp. 2382–2387. Piscataway, NJ: IEEE.
34. Solomonoff RJ. 1997 The discovery of algorithmic probability. *J. Comp. Syst. Sci.* **55**, 73–88. (doi:10.1006/jcss.1997.1500)
35. van Dijk SG, Polani DP. 2011 Grounding subgoals in information transitions. In *Adaptive Dynamic Programming and Reinforcement Learning* (*ADPRL*), *2011 IEEE Symposium, Paris, France, 11–15 April*, pp. 105–111. New York, NY: IEEE.
36. Doucet A, Godsill S, Andrieu C. 2000 On sequential Monte Carlo sampling methods for Bayesian filtering. *Stat. Comput.* **10**, 197–208. (doi:10.1023/A:1008935410038)
37. Geffner H. 2013 Computational models of planning. *Wiley Interdiscip. Rev. Cogn. Sci.* **4**, 341–356. (doi:10.1002/wcs.1233)
38. Doya K, Ishii S, Pouget A, Rao RPN (eds). 2007 *Bayesian brain: probabilistic approaches to neural coding*, 1 edn. Cambridge, MA: MIT Press.
39. Friston K. 2010 The free-energy principle: a unified brain theory? *Nat. Rev. Neurosci.* **11**, 127–138. (doi:10.1038/nrn2787)
40. Genovesio A, Tsujimoto S, Wise SP. 2012 Encoding goals but not abstract magnitude in the primate prefrontal cortex. *Neuron* **74**, 656–662. (doi:10.1016/j.neuron.2012.02.023)
41. Friston K, Samothrakis S, Montague R. 2012 Active inference and agency: optimal control without cost functions. *Biol. Cybern.* **106**, 523–541. (doi:10.1007/s00422-012-0512-8)
42. Lee D, Rushworth MFS, Walton ME, Watanabe M, Sakagami M. 2007 Functional specialization of the primate frontal cortex during decision making. *J. Neurosci.* **27**, 8170–8173. (doi:10.1523/JNEUROSCI.1561-07.2007)
43. Ortega PA, Braun DA. 2013 Thermodynamics as a theory of decision-making with information-processing costs. *Proc. R. Soc. A* **469**, 20120683. (doi:10.1098/rspa.2012.0683)
44. Anderson C, Boomsma JJ, Bartholdi JJ. 2002 Task partitioning in insect societies: bucket brigades. *Insectes Soc.* **49**, 171–180. (doi:10.1007/s00040-002-8298-7)
45. Hamann H, Karsai I, Schmickl T. 2013 Time delay implies cost on task switching: a model to investigate the efficiency of task partitioning. *Bull. Math. Biol.* **75**, 1181–1206. (doi:10.1007/s11538-013-9851-4)
46. Karsai I, Schmickl T. 2011 Regulation of task partitioning by a common stomach: a model of

nest construction in social wasps. *Behav. Ecol.* **22**, 819–830. (doi:10.1093/beheco/arr060)

47. Seeley TD, Visscher PK, Schlegel T, Hogan PM, Franks NR, Marshall JAR. 2012 Stop signals provide cross inhibition in collective decision-making by honeybee swarms. *Science* **335**, 108–111. (doi:10.1126/science.1210361)
48. Trianni V, Tuci E, Passino KM, Marshall JAR. 2011 Swarm cognition: an interdisciplinary approach to the study of self-organising biological collectives. *Swarm Intell.* **5**, 3–18. (doi:10.1007/s11721-010-0050-8)
49. Daw ND, Niv Y, Dayan P. 2005 Uncertainty-based competition between prefrontal and dorsolateral striatal systems for behavioral control. *Nat. Neurosci.* **8**, 1704–1711. (doi:10.1038/nn1560)
50. Pezzulo G, Rigoli F. 2011 The value of foresight: how prospection affects decision-making. *Front. Neurosci.* **5**, 79. (doi:10.3389/fnins.2011.00079)
51. Pezzulo G, Verschure P, Balkenius C, Pennartz C. 2014 The principles of goal-directed decision-making: from neural mechanisms to computation and robotics. *Phil. Trans. R. Soc. B* **369**, 20130470. (doi:10.1098/rstb.2013.0470)
52. Pezzulo G, van der Meer MA, Lansink CS, Pennartz CMA. 2014 Internally generated sequences in learning and executing goal-directed behavior. *Trends Cogn. Sci.* **18**, 647–657. (doi:10.1016/j.tics.2014.06.011)
53. Verschure P, Pennartz C, Pezzulo G. 2014 The why, what, where, when and how of goal directed choice: neuronal and computational principles. *Phil. Trans. R. Soc. B* **369**, 20130483. (doi:10.1098/rstb.2013.0483)
54. Barlow HB. 1961 Possible principles underlying the transformation of sensory messages. In *Sensory communication* (ed. WA Rosenblith). Cambridge, MA: MIT Press.
55. Kiebel SJ, Daunizeau J, Friston KJ. 2008 A hierarchy of time-scales and the brain. *PLoS Comput. Biol.* **4**, e1000209. (doi:10.1371/journal.pcbi.1000209)