ARTICLES
https://doi.org/10.1038/s41567-021-01170-x

nature physics

Check for updates

Topological limits to the parallel processing capability of network architectures

Giovanni Petri[1,2,8] ✉, Sebastian Musslick[3,8], Biswadip Dey[4], Kayhan Özcimder[5], David Turner[3], Nesreen K. Ahmed[6], Theodore L. Willke[6] and Jonathan D. Cohen[3,7]

The ability to learn new tasks and generalize to others is a remarkable characteristic of both human brains and recent artificial intelligence systems. The ability to perform multiple tasks simultaneously is also a key characteristic of parallel architectures, as is evident in the human brain and exploited in traditional parallel architectures. Here we show that these two characteristics reflect a fundamental tradeoff between interactive parallelism, which supports learning and generalization, and independent parallelism, which supports processing efficiency through concurrent multitasking. Although the maximum number of possible parallel tasks grows linearly with network size, under realistic scenarios their expected number grows sublinearly. Hence, even modest reliance on shared representations, which support learning and generalization, constrains the number of parallel tasks. This has profound consequences for understanding the human brain's mix of sequential and parallel capabilities, as well as for the development of artificial intelligence systems that can optimally manage the tradeoff between learning and processing efficiency.

There is a fundamental tension between two kinds of use of parallel distributed computing in network architectures. The first focuses on incorporating a variety of interacting constraints in the learning and processing of complex representations ('interactive parallelism'). This has been profitably exploited in theories of human cognitive function[1,2] and, most recently, in the design of 'deep learning' artificial systems[3–5]. The second kind of use, in contrast, focuses on the capacity of a network to carry out multiple processes independently ('independent parallelism'). This approach has been exploited by the massively parallel systems used in most modern computing clusters, and optimized by message-passing systems, such as MPI (message-passing interface)[6], that seek to identify and distribute independent components of computation.

Recent work has suggested that there is a fundamental tradeoff between these two types of parallelism that may help explain fundamental features of human cognitive function[7]. On the one hand, we can effortlessly perform many kinds of task at the same time, such as walking, talking and responding to our surroundings, all of which presumably involve extensive simultaneous computations. On the other hand, we are radically constrained in our ability to perform other kinds of task concurrently, such as planning a grocery list while simultaneously carrying out multidigit mental arithmetic. In cognitive psychology, this is attributed to a fundamental distinction between automatic and control-dependent processing[8,9]. The former is capable of effortless, simultaneous execution, while the latter is subject to seriality constraints on performance.

Early theorists proposed two alternative accounts for this constraint in control-dependent processing. One suggests that this reflects reliance on a centralized, limited capacity mechanism (akin to a central processing unit), thus explaining the dramatic limitation in the human ability to simultaneously perform multiple control-dependent tasks. The alternative interpretation suggests that constraints in control-dependent processing reflect the purpose, rather than a limitation, of control mechanisms, that is, to resolve conflicts that arise from competition among the resources required to perform specific combinations of tasks, which themselves rely on the shared use of representations[10–13].

Although compelling, the latter proposal was not undergirded by a formal analysis of the extent to which shared use of representations constrains processing at the system level. In particular, one concern might be that shared use of representations in a system as large as the human brain may pose minimal constraints on parallel processing. Recently, however, numerical work has shown that even modest sharing of representations among tasks can impose radical constraints on simultaneous execution due to crosstalk interference among tasks, and that the effects of such interference can be invariant to network size[14,15]. Understanding the source of such constraints, and explaining them explicitly in mechanistically and formally rigorous terms, remains an important challenge not only for understanding human performance—and how it arises from computations in the brain—but also for the design of artificial systems that can emulate human performance.

In this Article, we provide a formal analysis of the problem, based on a combination of graph theory and statistical mechanics of frustrated systems. We illustrate the mechanism by which even modest degrees of shared representations impose strong constraints on the number of tasks that can be performed simultaneously without the risk of interference from crosstalk between tasks. Our results highlight a fundamental tension in network architectures between the benefits that accrue from shared representations (that is, flexibility of processing and generalization[3–5]) and their cost in terms of processing efficiency (that is, the number of independent tasks that can be performed in parallel[7]).

Results

Measures of task dependency predict parallel processing capability in a trained neural network. To consider the problem of multitasking (that is, concurrent parallel processing) analytically, we

[1]ISI Foundation, Turin, Italy. [2]ISI Global Science Foundation, New York, NY, USA. [3]Princeton Neuroscience Institute, Princeton University, Princeton, NJ, USA. [4]Siemens Corporation, Princeton, NJ, USA. [5]MathWorks, Natick, MA, USA. [6]Intel Labs, Santa Clara, CA, USA. [7]Department of Psychology, Princeton University, Princeton, NJ, USA. [8]These authors contributed equally: Giovanni Petri, Sebastian Musslick. ✉e-mail: giovanni.petri@isi.it

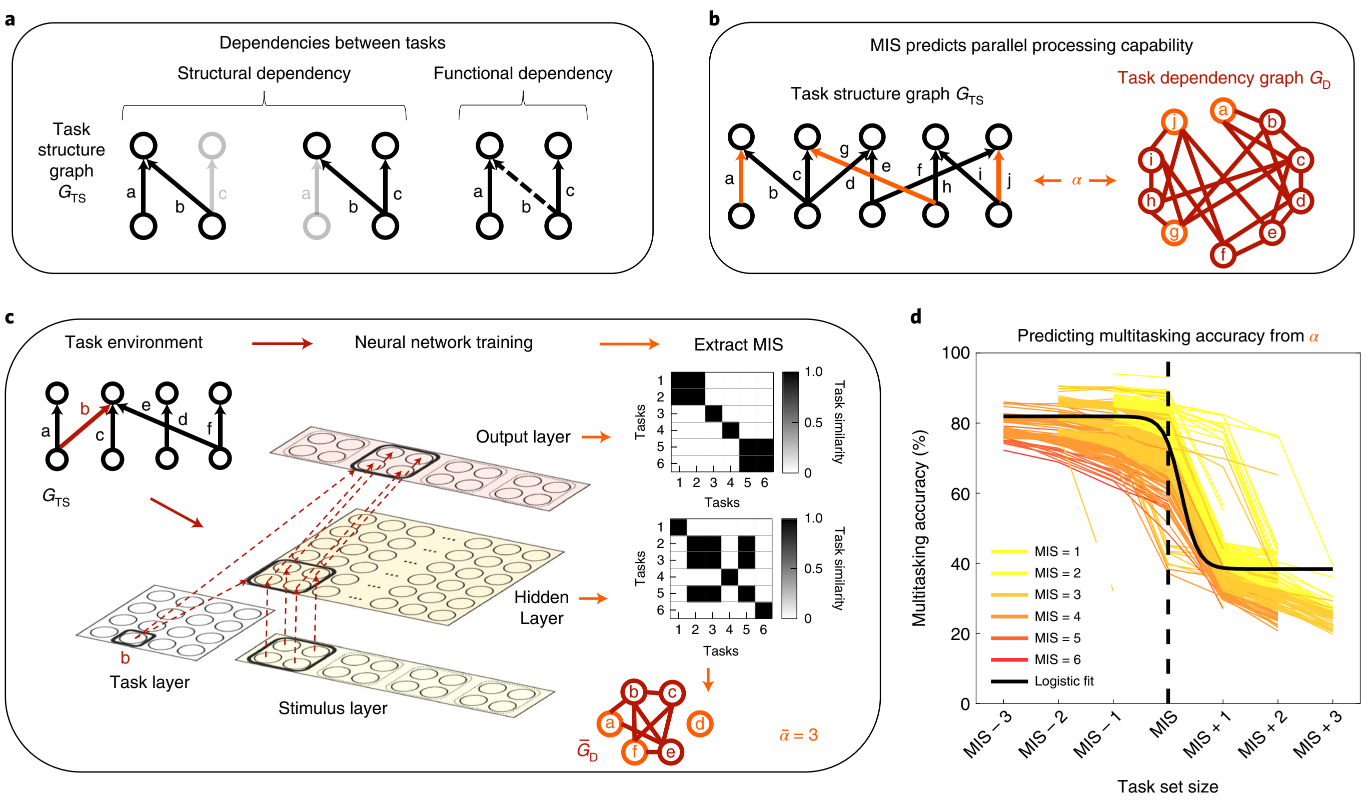

Fig. 1 | Graph-theoretic measures predict parallel processing capacity. a, A bipartite task structure graph, G_{TS}, describes tasks in terms of mappings from an input space to an output space. Each task corresponds to an edge in G_{TS} from an input node to an output node. Two tasks are (1) structurally dependent if they converge to the same output node (left) or originate from the same input node (middle) or (2) functionally dependent if their edges are connected by a third task (right). **b**, Dependencies between tasks in G_{TS} are expressed by the corresponding task dependency graph G_D: tasks are now represented as nodes, and they are connected if the two tasks are structurally or functionally dependent. The MIS of G_D corresponds to the largest set of tasks that a network can execute in parallel without interference[15]. Its cardinality, $\alpha = |\text{MIS}|$, is the maximal parallel processing capacity of the network G_{TS}. **c**, A neural network is trained on a set of tasks, in which each task requires the network to map a set of features from the stimulus layer via a hidden layer to a set of features on the output layer. Each task is designated by a unit in an additional (task) input layer that projects to both the hidden and output layers. All tasks in the environment can be expressed in terms of a task structure graph G_{TS} (as shown in **a**). The network is trained on all tasks by activating, on each trial, a particular task unit and an input unit corresponding to a stimulus feature in the set for that task, and requiring the network to activate the corresponding output unit. The average activity patterns at the hidden and output layers across all inputs under a given task are taken as the network's representation of that task. The two resulting similarity matrices (for the hidden and output layers) are used to infer dependencies between tasks based on shared task representations, and to construct the empirical task dependency graph $\bar{G}_D$, which we use to predict the empirical $\bar{\alpha}$. **d**, The parallel processing capacity of the trained network is predicted by $\bar{\alpha}$. We use the overbar notation to refer to quantities estimated from simulations. The plot shows the highest multitasking accuracy of the network as a function of the number of tasks it is asked to perform in parallel (performance curve) as indicated in relation to the network's MIS. Each line corresponds to the multitasking performance of a trained network, whereas the colour of each line indicates the predicted MIS for that network. The solid black line depicts the average fit of a logistic function to accuracy curves across networks.

first provide a formal definition of a task. More details are provided in Supplementary Section 1. Given an input space I of stimuli (for example, colours) and an output space O of responses (for example, verbal response), a task $T{:}\,I \to O$ represents a mapping between the two (for example, naming the colour of a stimulus), such that the mapping is independent of any other, and that selection of a feature from its input space can be made independently of any other. Different tasks can share an input space, output space or both (for example, reading a colour word such as 'red' out loud and naming the colour in which it is printed share an output space). When this occurs, there is the potential for the tasks to interfere with one another[14,16].

Such interference can be made explicit by describing the task structure in the form of a (bipartite) task structure graph $G_{TS} = G(\mathcal{I}, \mathcal{O}, \mathcal{T})$. G_{TS} makes the sharing of representations across tasks explicit (Fig. 1a), where $\mathcal{I}$, $\mathcal{O}$ and $\mathcal{T}$ are, respectively, the sets of input spaces, output spaces and tasks. A task $t \in \mathcal{T}$ is formally defined as mapping from an input space to an output space $t{:}\,I \to O$, with $I \in \mathcal{I}, O \in \mathcal{O}$. Whenever two tasks share an input node I or an output node O, we assume that they are at risk of interference due to direct crosstalk and therefore should not be executed in parallel; we call this dependency 'structural' because of the direct reliance on common resources[15]. Figure 1a depicts this type of dependency between tasks a and b and between b and c. Importantly, in addition to structural dependence, there can also be 'functional' dependence between two tasks: this is the case whenever, given two tasks, a third task maps the input space (that is, connects the input node) of the first task to the output space (that is, output node) of the second one. In Fig. 1a, tasks a and c are functionally dependent via task b, because activating a stimulus in task c's input space does the same for b, thus invoking a response to b that may conflict with the response to a. Finding the maximum number of tasks that can be simultaneously executed (that is, multitasked) is then equivalent to finding the largest set of edges in G_{TS} that are neither structurally nor functionally dependent on one another. In graph-theoretic terms, this corresponds to finding a maximum induced edge matching of G_{TS}: a subset of tasks in which none of the tasks either share a node or are connected by an edge. In Fig. 1b (left) we show an example of induced matching (in orange).

Interestingly, under an assumption that we will specify in detail shortly, all task dependencies can be made explicit in a derived graph, the task dependency graph G_{D}, in which nodes represent tasks and edges represent their (structural or functional) dependencies (Fig. 1b, right). Starting from G_{TS}, the dependency graph is built by considering the square of the line graph of G_{TS}. In fact, the line graph of G_{TS} encodes structural interferences between tasks. Taking its square corresponds to closing all open wedges and encodes functional interference. It can be shown that the maximum induced edge matching on G_{TS} corresponds to the maximum independent set (MIS) of G_{D} (ref. [17]), the largest set of nodes that are not connected by any edge. The cardinality of this set is called the independence number α of G_{D}.

This equivalence is key to our first main contribution: a neural network constrained to learn a task structure characterized by graph G_{TS} exhibits a maximum parallel capacity given by the independence number of the corresponding G_{D}.

To assess the correspondence of this theoretical measure of parallel processing capacity to the performance of an actual network, we trained a simple nonlinear feedforward network (Fig. 1c), with four layers, that has been used previously to simulate a wide array of empirical findings concerning human cognitive performance[18–20] (Methods). The network architecture entails two input layers, one that encodes the current stimulus (stimulus layer) and another that encodes the task to be performed on the stimulus (task layer). Both input layers project to a hidden layer that computes an internal representation of task-relevant input features of the stimulus. Finally, information encoded at the hidden layer is projected together with the task layer input to an output layer, at which the response of the network is computed. The weight projections from the task layer serve to bias processing towards task-relevant stimulus information represented at the hidden layer, as well as task-relevant responses at the output layer[18]. This, in turn, shapes the representations of the input and output space respectively for each task.

As a benchmark, we demonstrate the correspondence between the structure of G_{TS} and the one derived from the theoretical dependence graph G_{D} under the assumption of maximum sharing of representations. We train a set of networks to learn the mapping from inputs to outputs for each task, with fixed weight projections from the task layer to the hidden layer that are the same for tasks with shared input spaces. This guarantees the maximum amount of representation sharing between tasks. We refer to this as a minimal basis set representation, as it is the most compact form of representation at the hidden layer that can support the performance of all tasks. We trained 400 networks in this manner, varying the total number of tasks (between 4 and 30) and task structure graph G_{TS} (Fig. 1c). For each network trained on a task environment G_{TS}, we computed a theoretical task dependency graph, G_{D} (details are provided in the Methods and Supplementary Sections 2 and 3). Figure 1d shows that α predicts well the maximum number of tasks the network can perform in parallel. That is, the highest accuracy that the network can achieve (across all task combinations for a given task set size) drops as soon as the task set size exceeds α.

These results show that, when the network is constrained to learn maximally shared representations, the pattern of performance it exhibits is consistent with the task interference structure described by G_{TS}, and its parallel processing capacity can be accurately predicted from the corresponding G_{D}, which is easily obtained from G_{TS}. However, they do not address other network configurations that do not conform to the minimal basis set. Other weighting schemes are possible and are of interest for theoretical and practical reasons. In the Supplementary Information we show that our results are valid also for different fixed (Supplementary Section 4) and unconstrained (Supplementary Section 5) weights, as well as for more complex neural network architectures (Supplementary Section 6).

Maximum parallel capacity estimation for dependency graphs of arbitrary size. An important theoretical question that remains unanswered is how the relationship between the sharing of representations and the parallel processing capacity of a network scale with network size for very large networks, for which it is prohibitive to compute either α and $\bar{\alpha}$ directly, where we use the overbar notation for quantities estimated from simulations.

To address this problem, we develop a graph ensemble formulation of the MIS problem in terms of the degree distribution of the task dependency graph. This allows us to tease apart the roles of graph density and heterogeneity, independently of network size. To achieve this, we need to relate the MIS density ρ_α ($\rho_{\bar{\alpha}}$) to the degree distribution of G_{D} ($\bar{G}_{\mathrm{D}}$).

In the minimal basis set configuration, the degree distribution of G_{D} can be computed directly starting from G_{TS}. Because the task dependency graph G_{D} is the square of the line graph $\mathcal{L}(G_{\mathrm{TS}})$ of G_{TS}, the estimated degree $\bar{k}_e^{\mathrm{D}}$ of task e from input node I to output node O in G_{D} as $\bar{k}_e^{\mathrm{D}}$ (ref. [21]) is

$$\bar{k}_e^{\mathrm{D}} \simeq \frac{(k_{\mathrm{I}}-1)\langle k_{\mathrm{O}}^2\rangle}{\langle k_{\mathrm{O}}\rangle} + \frac{(k_{\mathrm{O}}-1)\langle k_{\mathrm{I}}^2\rangle}{\langle k_{\mathrm{I}}\rangle} - \frac{(k_{\mathrm{I}}-1)(k_{\mathrm{O}}-1)(\langle k_{\mathrm{I}}\rangle-1)(\langle k_{\mathrm{O}}\rangle-1)}{M-1} \qquad (1)$$

where $\langle\cdot\rangle$ are the expectation values of the outdegree k_{I} of the input node I and the indegree k_{O} of the output node O, and M is the number of edges in G_{TS} (or equivalently of nodes in G_{D}). Full details are provided in Supplementary Section 10.

In Fig. 2a we show that equation (1) gives good results (Pearson's $R > 0.9$, $P < 0.05$) for graphs of various densities and for various degree distributions. Note that $\bar{k}^{\mathrm{D}}$ is written in terms of the first two moments of P_{st}, the probability for an input node I with degree s to be linked to an output node O with degree t, recovering the previously observed connection between the heterogeneity of the G_{TS} graph and that of the corresponding dependency graph G_{D}.

When not in the minimal basis set scheme, it is not possible, in general, to obtain an expression for the degree distribution of $\bar{G}_{\mathrm{D}}$ from G_{TS}. However, this is a minor limitation as it is possible to estimate $\bar{G}_{\mathrm{D}}$ from the network activations, even when G_{TS} is unknown (Supplementary Section 4). Thus, going forward, we will focus exclusively on dependency graphs, disregarding their origin (theoretical or empirical). For simplicity of notation, we will denote these as G_{D} and their independence number as α, dropping the overbar notation.

To estimate the expected maximum independence number α, we build on recent work by Lucibello and Ricci-Tersenghi[22] and estimate the independence number density $\rho_\alpha = \alpha/M$ (where M is the number of nodes in G_{D}), based on a factor graph description of the maximum set packing problem, of which the independence number problem is a particular instance. Crucially, these expressions relate ρ_α to the graph's degree distribution, providing a window into the role of the network's topology.

Exploiting the properties of degree generating functions[23], the ρ_α estimate can be rewritten as

$$\rho_\alpha = \frac{\langle k\rangle}{\langle c\rangle}\left(1 - p_*^{c/(c-1)}\right) + \left(M_k(t) - M'(t)\right) \qquad (2)$$

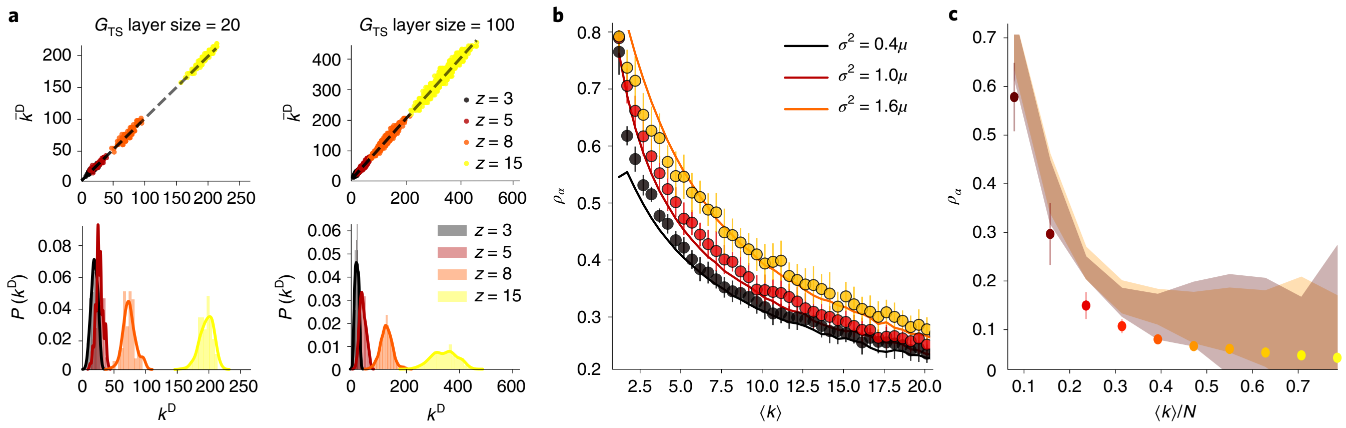

Fig. 2 | Graph-theoretic results for ρ_α. **a**, Comparison of the estimated $\bar{k}^D$ and actual k^D for dependency graphs obtained from task structure graphs with a range of average degrees z. For each value of z, the task dependency graph G_D was computed from its bipartite G_{TS} with a fixed degree on the input nodes and a binomial distribution of degrees for the output nodes (similar to an Erdös–Rényi graph, and following ref. [14]). The lower plot shows the same information in distribution form, with discrete bins corresponding to $\bar{k}^D$ and the solid line corresponding to the actual distribution. **b**, MIS densities ρ_α for a set of generic networks with Gaussian degree distributions of varying widths, comparing the exact computation (circles) with the values predicted from equation (2) (solid lines) as a function of the Gaussian distribution mean μ. The plots show that, for fixed μ, increasing the degree of heterogeneity (σ^2) is associated with increased ρ_α. **c**, MIS densities ρ_α for G_D as a function of the task structure graph density $\langle k \rangle/N$, comparing (1) the explicit calculation for theoretical G_D (circles); (2) the analytical results using equation (2) and the G_D degree distribution estimated using equation (1) (pink shading); (3) the analytical results using equation (2) with the measured G_D degree distribution. We created a set of G_{TS} graphs with binomial degree distributions on the input and output layers. For each parameter choice, we computed 50 G_{TS} graphs. Error bars on circles correspond to 1 s.d. of the resulting ρ_α values. The colours of the circles represent the average density of the corresponding G_D, which ranges from 0.2 to 0.7. The shaded areas are 1 s.d. intervals for the predicted ρ_α values.

where $t = \log p_*$, M' is the derivative of M with respect to t, and p_* needs to satisfy the self-consistent equation

$$p_* = \mathbb{E}_{\tilde{c}}\left(1 - \frac{1}{\langle k \rangle p_*} M'(t)\right)^{\tilde{c}} \tag{3}$$

Here k is the node degree in G_D, c and $\tilde{c}$ refer to the factor nodes' degrees and excess degrees, which in the case of the MIS are fixed to $c = 2$ and $\tilde{c} = 1$ (details are provided in Supplementary Section 8), $\mathbb{E}_{\tilde{c}}$ is the expection value over $\tilde{c}$ and $M_k(t)$ is the generating function for the degree distribution $p(k)$.

For classes of graphs that have analytical degree generating functions, it is possible to obtain insights into the role of the density and heterogeneity of G_D directly. For example, for a Gaussian distribution with mean $\langle k \rangle$ and variance σ^2, the moment generating function takes the form $M_k(t) = e^{\langle k \rangle t + \sigma^2 t^2/2}$. Substituting the expression above, we obtain

$$\rho_\alpha = \frac{\langle k \rangle}{\langle c \rangle}\left(1 - p_*^{c/(c-1)}\right) + M_k(\ln p_*)(1 - \langle k \rangle - \sigma^2 \ln p_*) \tag{4}$$

In Fig. 2b we show that this expression provides a close approximation of the behaviour of ρ_α for increasing network density and for various levels of degree heterogeneity of G_D. Importantly, it provides an analytical grounding for the previous empirical observations that increased heterogeneity of task overlap for a given average density results in a higher ρ_α (refs. [14,15]). Here, we use Gaussian degree distributions to explicitly illustrate the impact of the density of the dependency graph, which depends, in turn, on the density of the task structure graph and its degree heterogeneity: for a fixed size, dense and uniform graphs have a smaller MIS than sparse, heterogeneous ones.

Finally, we show that it is possible to predict ρ_α starting from the degree distribution of G_{TS}, estimating the degree distribution of G_D from it, and then plugging it into equation (2). We computed this for a set G_{TS} with fixed number of nodes per layer, N, and for increasing densities (Supplementary Figs. 9 and 10 show other G_{TS} topologies). We find that the prediction obtained from the estimated and actual degree distributions of G_D are in agreement, and they both yield a strict upper bound on ρ_α.

Effective parallel processing capacity. The independence number specifies the maximum capacity and is specific to a particular subset (or very few subsets) of tasks. Thus, the independence number does not address the more practical question: what is the greatest number of tasks that the system is expected to perform simultaneously on average, given a probability distribution of tasks in the environment. In other words, given a task set T of cardinality $|T| = \gamma$, what is the probability P_γ that those tasks are both available for execution and can be successfully executed at the same time?

The probability P_γ is a special case of the probability $P(\theta;\gamma,G_D)$ of successfully executing θ out of γ tasks from a dependency graph G_D. The latter requires the θ nodes in $T \subset G_D$ not to be linked with each other, and the remaining $\gamma - \theta$ nodes to be connected to at least one of the first θ tasks. For a graph G_D, we can estimate the probability of successfully executing $1 \le \theta \le \gamma$ tasks in T as

$$P(\theta;\gamma,\mathcal{G}_D) \simeq \left(1 - \frac{\langle k^2 \rangle}{2M_D}\right)^{\binom{\theta}{2}} \left(\frac{\theta \langle k \rangle^2}{2M_D}\right)^{\gamma-\theta} \tag{5}$$

where M_D, $\langle k \rangle$ and $\langle k^2 \rangle$ are, respectively, the number of edges in G_D and the first and second moments of G_D's degree distribution (Supplementary Section 9).

In Fig. 3a we show the value of P_γ as a function of γ, for various G_D with variable network size M but fixed network density. Naturally, for $\gamma = 1$, the probability of executing the task is always 1 because a single task cannot interfere with itself, but P_γ decreases very rapidly as the number of attempted tasks increases and, remarkably, does not depend on the network size (details are provided in Supplementary

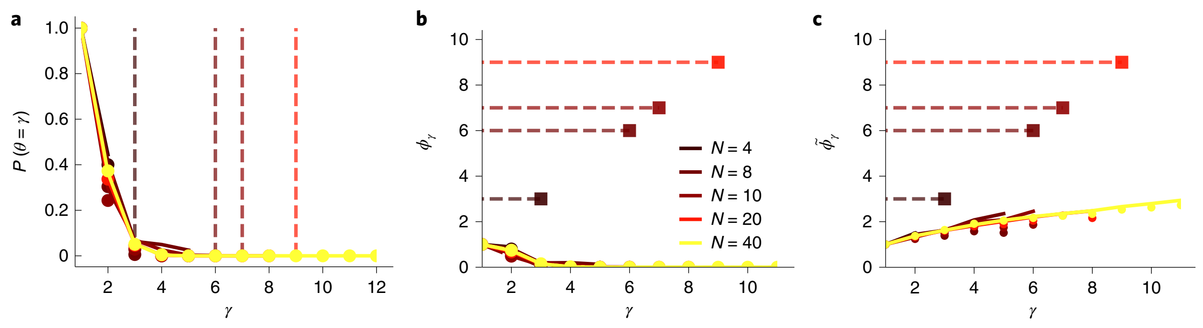

Fig. 3 | Graph-theoretic results for P_γ, ϕ_γ and $\tilde{\phi}_\gamma$. **a**, Probability P_γ that all tasks in a task subset of cardinality γ are performed successfully (that is, are independent). We plot P_γ measured directly (circles) in dependency graphs obtained from Erdös–Rényi task graphs with density $\rho = 0.2$ and variable size ($N = 4, 8, 10, 20, 40$) and compare them with the predictions of equation (5) (solid lines). The theoretical P_γ values are slightly higher than the measured values, but overall we find good agreement between the two. Vertical dashed lines highlight the MIS values for various N and show how, for all sizes, the probability of randomly choosing a maximal independent task set is vanishingly small. **b**, The ϕ_γ values (obtained using the P_γ in **a**) are much lower than the MIS for the corresponding size (shown as the horizontal dashed lines); moreover, ϕ_γ displays little dependence on the network size N, as opposed to the MIS, which grows linearly instead. **c**, Even under the more permissive reward function, $\tilde{\phi}_\gamma$ only grows sublinearly with γ and again shows no dependence on G_{TS} size N, leading to strongly diminishing returns for the effective parallel capacity.

Section 9). Equation (3) confirms analytically the size independence of the MIS previously observed in numerical experiments by Feng et al.[14]—at fixed density for G_D, $\langle k\rangle$, $\langle k^2\rangle$ and M_D all scale as M^2, making equation (3) independent of M.

To quantify how the rapid decrease in P_γ relates to performance, we associate a reward with each multitasking attempt. We consider two reward schemes: (1) in the 'all-or-nothing' scheme, we give a positive reward to an attempt to perform γ tasks only if all tasks are successful (that is, independent in G_D) and no reward otherwise; (2) in the 'graded' scheme, we give a reward to each multitasking attempt on γ tasks proportional to the maximum number of independent tasks $\gamma' \le \gamma$ in that set.

These schemes encode two extremes in how rewards for performance might depend on multitask success. The all-or-nothing scheme corresponds to situations in which the outcomes of the tasks can influence one another, and thus all tasks need to be successfully performed (for example, juggling a collection of objects requires all individual objects to be successfully juggled; failing on one is likely to induce failure on the others). By contrast, the graded scheme corresponds to situations in which task outcomes are not correlated (for example, driving and listening to a conversation) and hence failing one task does not induce failure of others. For the former, the expected reward is therefore written, modulo a multiplicative coefficient, as

$$\phi(\gamma, G_D) = \gamma P_\gamma + 0(1-P_\gamma) = \gamma P_\gamma \tag{6}$$

which peaks at low values of γ and rapidly converges to zero (Fig. 3b).

For the more permissive graded scheme, we have

$$\tilde{\phi}(\gamma, G_D) = \sum\nolimits_{\theta=1}^{\gamma} P(\theta;\gamma,G_D)\theta \tag{7}$$

In this case, $\tilde{\phi}$ grows for increasing γ values (Fig. 3c). This is expected, because, under this scheme, for any task subset the reward is positive (for example, at the limit if the task set is the whole network, the reward is proportional to the MIS size). Despite this, the average reward $\tilde{\phi}$ grows sublinearly with γ and, again, does not depend on the network size. As a consequence of this sublinear increase, any increase in dependency graph size is associated with diminishing returns in both ϕ and $\tilde{\phi}$. In the Supplementary Information (Supplementary Figs. 12 and 13), we show that, in the case of fixed average degree of G_D, the effective parallel processing capacity weakly depends on M, but the qualitative results do not change. Finally, we show that, also in the small M limit (outside the regime of validity of the formal treatment), a qualitatively similar effect can be observed for the empirical effective parallel processing capacity of trained neural networks (Supplementary Figs. 7–14).

Discussion

The work presented provides a formal analysis of the idea that the two forms of parallelism described here are not merely differences in computational strategy, but reflect a fundamental computational tradeoff in network architectures between efficiency of learning and generalization versus efficiency of processing: the very network fabric that supports interactive parallelism by sharing representations between tasks (for example, for learning and/or generalization) induces limits on independent parallelism and processing efficiency—that is, their ability to perform multiple tasks simultaneously[7]. Here, we have presented an approximation of the problem that is analytically tractable and thus can be used to examine it at arbitrary scales, permitting an analysis of its manifestation in more complex systems—both natural and artificial. Formally, we were able to summarize how the topology of the task dependency graph affects the neural architecture's parallel capacity and observed that the benefits of increases in network size scale in a strikingly sublinear manner, with rapidly decreasing returns in parallel capacity for larger networks, even when the proportion of shared representations (and attendant rate of competition) is kept constant. Empirically, we have validated a parsimonious method to estimate the underlying task dependency graph from individual patterns of task-specific activity, which can be extracted from data (for example, from neural data and neural network activations).

Although the network models we used have direct mappings between inputs and outputs, the definition of a task used in our analyses applies in deeper networks as well, as each pair of layers can be considered as an input–output mapping and thus the entire network can be considered as a series of such single-layered networks. From this perspective, a task to be executed by the network as a whole can be decomposed into a series of subtasks, traversing the various layers. Although this allows for the task to be successfully reproduced by multiple paths, at the same time the likelihood of interference between pathways implementing different tasks increases with the number of intermediate layers (that is, opportunities for intersection), compounding the effects we have described for single-layer networks[24,25]. The same logic applies to recurrent networks. These factors are similar to the effects of path structure on controllability in unfolded temporal graphs[26,27].

Our work also provides the basis for developing methods of assessing the parallel processing capacity of natural agents (for example, humans) for a given set of tasks. Previously proposed methods have used explicit signal modelling to infer the parallel processing capacity of a system from behavioural data (reaction time distributions) generated by actual task performance[28–30]. Our method complements these by providing a means for estimating parallel processing capability when the underlying task structure is unknown using the number of measurements linear in the number of tasks, as opposed to the factorial number required by previous methods. This may be valuable for important real-world domains, where multitasking is critical, but in which it is impractical to individually and exhaustively evaluate all of the potential task combinations—for example, pilots monitoring a large number of instruments.

One potential limitation of the proposed analysis is that averages of task representations need not necessarily reflect the extent to which sharing of representations occurs across tasks, which must occur at the level of individual stimulus features. Although the results presented in this, as well as other work[15,31], suggest that task averages do seem to provide a good proxy for the similarity in representations across tasks, others have reported work on using geometric measures to identify the manifolds on which representations for different tasks live in neural networks[32,33]. It remains a matter for future research to explore how well these measures can be used to predict the parallel processing capacity of network architectures.

Finally, at a higher level of analysis, our methods may also help shed light on how a system balances the efficiency of learning and generalization provided by interactive parallelism and shared representations, at the cost of serial processing, with the efficiency of processing provided by independent parallelism, at the cost of greater training time and task specificity.

Online content

Any methods, additional references, Nature Research reporting summaries, source data, extended data, supplementary information, acknowledgements, peer review information; details of author contributions and competing interests; and statements of data and code availability are available at https://doi.org/10.1038/s41567-021-01170-x.

Received: 1 April 2019; Accepted: 8 January 2021;
Published online: 18 February 2021

References

1. McClelland, J. L., Rumelhart, D. E. & Hinton, G. E. *The Appeal of Parallel Distributed Processing* (MIT Press, 1986).
2. Rogers, T. T. & McClelland, J. L. *Semantic Cognition: A Parallel Distributed Processing Approach* (MIT Press, 2004).
3. Bengio, Y., Courville, A. & Vincent, P. Representation learning: a review and new perspectives. *IEEE Trans. Pattern Anal. Mach. Intell.* **35**, 1798–1828 (2013).
4. Caruana, R. Multitask learning. *Mach. Learn.* **28**, 41–75 (1997).
5. Baxter, J. Learning internal representations. In *Proc. Eighth Annual Conference on Computational Learning Theory* 311–320 (ACM, 1995).
6. Gropp, W., Lusk, E., Doss, N. & Skjellum, A. A high-performance, portable implementation of the MPI message passing interface standard. *Parallel Comput.* **22**, 789–828 (1996).
7. Musslick, S. et al. Multitasking capability versus learning efficiency in neural network architectures. In *Proc. 39th Annual Meeting of the Cognitive Science Society* 829–834 (Cognitive Science Society, 2017).
8. Posner, M. I. & Snyder, C. R. in *Information Processing and Cognition: The Loyola Symposium* 55–85 (Erlbaum, 1975).
9. Shiffrin, R. M. & Schneider, W. Controlled and automatic human information processing: II. Perceptual learning, automatic attending and a general theory. *Psychol. Rev.* **84**, 127–190 (1977).
10. Wickens, C. D. in *Multiple-Task Performance* 1st edn (ed. Damos, D. L.) Ch. 1 (CRC Press, 1991).
11. Allport, D. A. in *New Directions in Cognitive Psychology* (ed. Claxton, G. L.) 112–153 (Routledge, 1980).
12. Meyer, D. E. & Kieras, D. E. A computational theory of executive cognitive processes and multiple-task performance: Part I. Basic mechanisms. *Psychol. Rev.* **104**, 3–65 (1997).
13. Navon, D. & Gopher, D. On the economy of the human-processing system. *Psychol. Rev.* **86**, 214–255 (1979).
14. Feng, S. F., Schwemmer, M., Gershman, S. J. & Cohen, J. D. Multitasking vs. multiplexing: toward a normative account of limitations in the simultaneous execution of control-demanding behaviors. *Cogn. Affect. Behav. Neurosci.* **14**, 129–146 (2014).
15. Musslick, S. et al. Controlled vs. automatic processing: a graph-theoretic approach to the analysis of serial vs. parallel processing in neural network architectures. In *Proc. 38th Annual Meeting of the Cognitive Science Society* 1547–1552 (Cognitive Science Society, 2016).
16. Stroop, J. R. Studies of interference in serial verbal reactions. *J. Exp. Psychol.* **18**, 643–662 (1935).
17. Gavril, F. Algorithms for a maximum clique and a maximum independent set of a circle graph. *Networks* **3**, 261–273 (1973).
18. Cohen, J. D., Dunbar, K. & McClelland, J. L. On the control of automatic processes: a parallel distributed processing account of the stroop effect. *Psychol. Rev.* **97**, 332–361 (1990).
19. Cohen, J. D., Servan-Schreiber, D. & McClelland, J. L. A parallel distributed processing approach to automaticity. *Am. J. Psychol.* **105**, 239–269 (1992).
20. Botvinick, M. M., Braver, T. S., Barch, D. M., Carter, C. S. & Cohen, J. D. Conflict monitoring and cognitive control. *Psychol. Rev.* **108**, 624–652 (2001).
21. Newman, M. *Networks: An Introduction* (Oxford Univ. Press, 2010).
22. Lucibello, C. & Ricci-Tersenghi, F. The statistical mechanics of random set packing and a generalization of the Karp–Sipser algorithm. *Int. J. Stat. Mech.* **2014**, 1–13 (2014).
23. Newman, M. E. J. Random graphs with clustering. *Phys. Rev. Lett.* **103**, 058701 (2009).
24. Alon, N. et al. A graph-theoretic approach to multitasking. In *Proc. 31st Annual Conference on Neural Information Processing Systems* 2101–2110 (NIPS, 2017).
25. Alon, N. et al. Multitasking capacity: hardness results and improved constructions. *SIAM J. Discrete Math.* **34**, 885–903 (2020).
26. Pósfai, M. & Hövel, P. Structural controllability of temporal networks. *New J. Phys.* **16**, 123055 (2014).
27. Li, A., Cornelius, S. P., Liu, Y.-Y., Wang, L. & Barabási, A.-L. The fundamental advantages of temporal networks. *Science* **358**, 1042–1046 (2017).
28. Townsend, J. T. & Wenger, M. J. A theory of interactive parallel processing: new capacity measures and predictions for a response time inequality series. *Psychol. Rev.* **111**, 1003–1035 (2004).
29. Townsend, J. T. & Wenger, M. J. The serial–parallel dilemma: a case study in a linkage of theory and method. *Psychon. Bull. Rev.* **11**, 391–418 (2004).
30. Wenger, M. J. & Townsend, J. T. On the costs and benefits of faces and words: process characteristics of feature search in highly meaningful stimuli. *J. Exp. Psychol. Human* **32**, 755–779 (2006).
31. Musslick, S. & Cohen, J. D. A mechanistic account of constraints on control-dependent processing: shared representation, conflict and persistence. In *Proc. 41st Annual Meeting of the Cognitive Science Society* 849–855 (Cognitive Science Society, 2019).
32. Bernardi, S. et al. The geometry of abstraction in hippocampus and prefrontal cortex. *Cell* **183**, 954–967 (2020).
33. Cohen, U., Chung, S. Y., Lee, D. D. & Sompolinsky, H. Separability and geometry of object manifolds in deep neural networks. *Nat. Commun.* **11**, 746 (2020).

Methods

Neural network architecture and processing. We used a standard nonlinear feedforward network, with four layers, that has been used previously to simulate a wide array of empirical findings concerning human cognitive performance[2,18,20]. The network consists of two input layers, one of which represents the stimulus presented to the network and another that encodes an instruction for the task that the network has to perform on this stimulus. Both input layers project to a hidden layer. Unless stated otherwise, the hidden layer contained 100 units. Both the hidden layer and the task layer further project to an output layer that computes the network's response. The real-valued activity of each input unit constitutes the current stimulus. Activated units in the task layer indicate the task(s) to be currently executed. Performing a single task corresponds to clamping the corresponding task unit to 1 (activated) while all other units are set to 0. Multitasking conditions are represented by activating multiple task units at the same time. Units in the hidden and output layers take values between 0 and 1, as determined by a logistic activation function applied to their net input. Stimulus input units are structured according to D dimensions (subvectors of the stimulus pattern), each of which is comprised of a set of D feature units with only one feature unit activated per dimension. Similarly, output units are organized into D response dimensions, with only one of the D response units permitted to be active within a response dimension. Each task is represented by a single task input unit that is associated with a set of unique, one-to-one mappings between the input units in one stimulus dimension and the output units in one response dimension, and that is independent of the mappings for all other tasks (Fig. 1c). The number of stimulus input dimensions and response dimensions was varied between four and nine across environments. The task mappings were generated with the Erdös–Rényi model such that the number of overlapping tasks for a given stimulus input dimension z varied between one and seven. For each environment G_{TS}, a network was initialized with a set of small random weights and then trained using the backpropagation algorithm[34] to produce the task-specified response for all stimuli in each task until it reached a mean-squared error performance of 0.001. (The training criterion was chosen such that the network achieves single task performance comparable to that of human participants on tasks requiring simple stimulus–response mappings (accuracy %).) We constrained the learned representations of the network to reflect the task similarity structure of the environment G_{TS} by fixing the weights from the task units to the hidden layer: weight vectors for tasks relying on the same stimulus input dimensions were set to yield a Pearson correlation coefficient of value 1 whereas weight vectors for tasks of non-overlapping stimulus dimensions were uncorrelated.

Dependency graph extraction. We followed the analysis described in ref. [15] and focused on the representations (patterns of activity) over the hidden and output units, insofar as these reflect the computations carried out by the network required to perform each task. In particular, we are interested in the characteristics of these representations for each task, how they compare across tasks, and how these factors correspond to empirically evaluated parallel processing performance. The representations associated with each task can be characterized by calculating, for each unit in the hidden and output layers, the mean of its activity over all of the stimuli for a given task. This mean pattern of activity can then be used as a representation of the task.

Correlating patterns of activity within a layer across tasks yields a task similarity matrix that can be examined separately for the hidden and output layers of the network. This can then be used to assess the extent to which different tasks rely on similar or different representations within each layer of the network. Figure 1c provides an example of such similarity matrices (thresholded for similarity correlations above $\theta = 0.5$). Tasks that have similar representations over the hidden layer can be inferred to rely on the same input dimension—that is, they share an input component in the bipartite graph representation of the network—and tasks that are similar at the output layer can be inferred to share an output component. Accordingly, a task dependency graph $\bar{G}_{\mathrm{D}}$ (of the type shown in Fig. 1b) can be constructed by measuring the patterns of activity observed in the network while it performs each individual task.

Assessing multitasking accuracy. To test the overall multitasking performance for each network, we considered all sets of 'multitaskable' tasks on which it was trained; that is, all sets of structurally independent tasks for which each task had input and output dimensions that were distinct from all of the others in the set. The accuracy of the network on a single task was determined by the probability of responding correctly in the task-relevant output dimension, averaged across all stimuli. Multitasking accuracy for a given set of tasks was determined by the average probability of responding correctly across all task-relevant output dimensions, averaged across all stimuli. The probability of responding correctly in a given output dimension was determined by a leaky competitive accumulator layer[35] (optimized for performance under each multitasking condition), implementing the assumption that the network could only provide one response per response dimension (details are provided in the Supplementary Information). To statistically assess the predictability of multitasking performance, we fit a logistic curve to the best multitasking accuracy as a function of set task set size. To avoid ill-conditioned solutions for logistic fits, we excluded networks for which the number of data points fell below three. Multitasking accuracy is considered well predicted if the inflection point (bias) of the fitted logistic lies above the predicted task set size and below the predicted set size + 1.

Statistical evaluation of α prediction from neural network simulations. To statistically evaluate the MIS prediction in Fig. 1d, we fit a logistic function to the accuracy of a network's performance as a function of set size. We find that the inflection point of the sigmoid curve is accurately predicted by the α derived from G_{D}. That is, the inflection point (that is, offset) of the curve lies significantly above a set size equal to α, $t(352) = 9.1465$, $P < 1 \times 10^{-17}$, and below a set size of $\alpha + 1$, $t(352) = -24.3986$, $P < 1 \times 10^{-77}$. These predictions turn out to be robust for a range of different performance metrics, number of hidden units in the network, as well as choices of θ used to extract $\bar{G}_{\mathrm{D}}$ (Methods and Supplementary Section 3).

Data availability

Example data files are available at https://github.com/lordgrilo/Multitasking_capacity. Source data are provided with this paper.

Code availability

Code to reproduce the simulations and analysis reported here is availabile at https://github.com/lordgrilo/Multitasking_capacity.

References

34. Hinton, G. E., Rumelhart, D. E. & Williams, R. J. Learning representations by back-propagating errors. *Nature* **323**, 533–536 (1986).
35. Usher, M. & McClelland, J. L. The time course of perceptual choice: the leaky, competing accumulator model. *Psychol. Rev.* **108**, 550–592 (2001).

Acknowledgements

G.P. has received funding support from Fondazione Compagnia San Paolo and from Intesa Sanpaolo Innovation Center. S.M. and J.D.C. acknowledge support from the John Templeton Foundation. The opinions expressed in this publication are those of the authors and do not necessarily reflect the views of the John Templeton Foundation.

Author contributions

G.P., S.M., B.D., K.Ö., N.K.A., T.L.W. and J.D.C. designed the research. G.P. developed and performed analytical and numerical calculations. S.M. and D.T. designed, implemented and performed the neural network simulations. S.M., K.Ö., B.D. and N.K.A. provided tools and performed neural network analysis. J.D.C. and T.L.W. conceptualized research and provided advice for all parts of the work. G.P., S.M., B.D., K.Ö., N.K.A., T.L.W. and J.D.C. wrote the manuscript.

Competing interests

The authors declare no competing interests.

Additional information

Supplementary information The online version contains supplementary material available at https://doi.org/10.1038/s41567-021-01170-x.

Correspondence and requests for materials should be addressed to G.P.

Peer review information *Nature Physics* thanks Hartmut Lentz and the other, anonymous, reviewer(s) for their contribution to the peer review of this work.

Reprints and permissions information is available at www.nature.com/reprints.