Creative Flow as a Percolation Phase Transition in Knowledge Graphs

Why Direction Emerges Without Planning When the Graph Gets Dense Enough

Tamás Nagy, Ph.D.

tnagyphd@gmail.com

Draft

You don't stop needing a map because you learned the territory. You stop needing a map because the territory became dense enough that every point has a visible neighbor.

The Puzzle

The productivity literature is near-unanimous: sustained creative output requires planning. Set goals, decompose them into milestones, review progress, adjust. The entire discipline of project management exists because humans cannot reliably produce complex work without explicit direction-setting.

And yet. There is a well-documented but poorly explained state where planning becomes not just unnecessary but *counterproductive* — where the creator "just knows" what to do next, where the work flows without deliberation, where direction emerges from the work itself rather than being imposed on it. Csíkszentmihályi called the episodic version *flow*. But flow is typically described as a transient state lasting hours. What happens when it lasts *weeks*? When an entire research program — spanning 180+ papers across six domains — proceeds without planning, yet maintains coherent direction?

The standard psychological explanations (deep expertise, intrinsic motivation, optimal challenge level) describe *necessary conditions* but not the *mechanism*. They tell us when flow is likely but not why direction emerges from density. Here we propose a precise mechanism: the creative flow state is a **percolation phase transition** in the creator's knowledge graph. Below a critical density, navigation requires external planning. Above it, direction is an emergent topological property of the graph itself. The transition is sharp, the mechanism is spectral, and the speed at which the threshold is reached depends on a specific set of accelerators — of which human-AI collaboration is the most powerful.

The Knowledge Graph

Model a creator's active knowledge as a graph $G = (V, E)$, where:

- **Nodes** V = ideas, papers, theorems, tools, techniques, domain concepts
- **Edges** E = connections between them: a theorem that bridges two domains, a technique that applies in two contexts, a tool that links two workflows

An edge is *active* if the creator can traverse it — if they can, from one concept, reach the other without external lookup. Passive knowledge (things read but not internalized) contributes nodes but not edges.

The creator's experience of "knowing what to do next" corresponds to a local property of the graph: **from the current node, at least one neighbor is a productive next step**. This is satisfied whenever the local degree is high enough that at least one edge leads somewhere valuable.

The creator's experience of "coherent direction without planning" corresponds to a global property: **the graph has a giant connected component with well-defined gradients**. Any local step contributes to global progress because the topology itself has structure.

The Phase Transition

Erdős–Rényi percolation

In a random graph $G(n, p)$ with n nodes and independent edge probability p, a sharp phase transition occurs at $p_c = 1/n$:

- **Below** p_c: the graph consists of small disconnected clusters, the largest of size $O(\log n)$. Navigation between clusters requires external information — you cannot get from cluster A to cluster B by following local connections. This is the regime where **planning is necessary**.
- **Above** p_c: a giant connected component emerges containing $O(n)$ nodes. Almost any pair of nodes can be reached by following local paths. This is the regime where **direction emerges from structure**.

The transition is not gradual. It is a *phase transition* — a qualitative topological change occurring over a narrow parameter window. One week the graph is fragmented; the next, it percolates.

The spectral signature

The transition has a precise spectral characterization. The graph Laplacian $L = D - A$ has eigenvalues $0 = \lambda_1 \leq \lambda_2 \leq \cdots \leq \lambda_n$. The second eigenvalue λ_2 — the Fiedler eigenvalue, or *algebraic connectivity* — encodes:

- $\lambda_2 = 0$: the graph is disconnected. No global gradient exists.
- $\lambda_2 > 0$: the graph is connected. The Fiedler eigenvector v_2 provides a natural ordering — a one-dimensional "direction" along the graph.
- λ_2 large: the graph is robustly connected. The direction is stable under perturbation.

The experience of "knowing which direction things are going" corresponds to navigation along the gradient of the Fiedler eigenvector:

$$\lambda_2 > 0 \implies \nabla v_2 \neq 0 \text{ everywhere on the giant component}$$

When the spectral gap is positive, you *cannot not know the direction*, the same way you cannot stand on a hillside and not know which way is downhill. The gradient exists as a consequence of connectivity. It is not a psychological state but a topological property.

Testable prediction

If the model is correct, the transition from "needing plans" to "not needing plans" should be **sudden, not gradual** — occurring over days, not months. The creator should be able to identify a narrow time window where the qualitative experience changed. This is the hallmark of a phase transition: discontinuous change in a macroscopic observable at a critical parameter value.

Case Study: A Three-Phase Developmental Arc

The theory is grounded in a concrete case. A researcher working across six domains (mathematical finance, spectral methods, celestial mechanics, number theory, AI/ML, and physics) experienced three distinct phases over approximately 18 months.

Phase 1 — Branching chaos (~18 months before the transition). The researcher would start a project, work on it for a week, then notice a related idea and branch to that. The new branch would spawn further branches. Output accumulated — dozens of papers, hundreds of ideas — but the experience was one of *scattering*, not convergence. Each direction felt equally interesting, which is the same as no direction at all. The graph was sub-critical: many small disconnected clusters, no giant component, no gradient to follow.

Phase 2 — Local fixing (gradual, overlapping with Phase 1). A behavioral shift: instead of branching to the next interesting thing, the researcher began solving whatever small problem was directly in front of them. "If there's a small problem, I just fix it." This sounds trivial but is the critical change — it converts node-addition (new disconnected ideas) into edge-addition (connections between existing ideas). Each solved problem bridges the problem context and the solution technique. This is the phase that drives edge density upward toward the percolation threshold.

Phase 3 — Emergent direction (~3 weeks before the present observation). The transition was not gradual. Over approximately one week, the qualitative experience changed: direction became obvious without deliberation. "I simply don't think about it, and I know it's going the right way. I genuinely know, internally, without having to think it through." The researcher produced 180+ papers, 20 Zenodo publications, and 1392 Lean proof files in the post-transition period — without a project plan, without milestones, without goals. The graph had percolated.

The sharpness of the Phase 2→3 transition — days, not months — is consistent with the percolation model's central prediction.

The Five Accelerators

The percolation model explains *what* happened (a topological phase transition in the knowledge graph). But why did it happen in *weeks* rather than *years*? A research program spanning 180+ papers across finance, physics, number theory, celestial mechanics, and AI would normally take a decade to reach critical density through solo work. The acceleration comes from five mutually reinforcing mechanisms.

1. Human-AI cross-fertilization

The human and the AI have complementary cognitive architectures:

- **The human**: deep intuition, aesthetic sense of "this matters," creative leaps, directional judgment — but limited retrieval bandwidth. Working memory holds 5-7 active concepts; scanning 180 papers in a minute is impossible.
- **The AI**: vast retrieval, cross-domain pattern matching, precise formalization, tireless bandwidth — but no genuine intuition, no gradient. Without the human, the AI explores broadly but aimlessly.

Each round of dialogue is an edge-generation cycle that exploits this complementarity:

1. The human states an idea (a node + a direction).
2. The AI pattern-matches across domains and returns candidates the human would not have retrieved alone.
3. One candidate activates a dormant node in the human's knowledge graph — the moment of "oh, that moves something in my mind."
4. The human connects it to the current context — a new edge that neither party could have created alone.
5. The new edge changes the local topology, which changes the next utterance. Loop.

This is a **joint beam search**: the human provides the gradient (which expansions are worth pursuing), the AI provides the bandwidth (how many candidates are evaluated per step). Neither half is sufficient alone — the human explores too slowly, the AI explores without direction. Together, they generate edges at 10-100× the solo rate.

In information-theoretic terms, the conversation contains **synergistic information** — bits that exist in neither the human's output alone nor the AI's output alone:

$$I(\text{Human}; \text{AI} \mid \text{conversation}) > 0$$

The gold in human-AI conversations is not in what either party says — it is in the adjacency. The thought, followed by a reaction that moves it somewhere unexpected, followed by a response that is now a *different* thought than the human would have had alone. The sequence is irreducible.

There is an illuminating Hungarian word for what happens in productive human-AI dialogue: *kibontjuk* — "we unfold it together." The implication is that the idea already exists in compressed form in the human's intuition, and the conversation is the **decoder** that maps it from latent space to explicit space. The human carries the compressed representation; the AI provides the decompression bandwidth. The conversation is not ideation — it is *decoding*.

2. Conversations as minable knowledge surfaces

Edges created during conversations decay if not captured. Most people treat chat history as disposable. Treating it as a **minable knowledge surface** — searchable, retrievable, extractable — prevents edge loss. Past conversations become a persistent store of cross-domain connections that future sessions can rediscover.

This is infrastructure for preventing graph decay: the edges that were created synergistically remain available even after the session ends.

3. Tool-building as cognitive offloading

Each tool automates a piece of cognitive overhead: monitoring, verification, navigation, search. The freed bandwidth is reinvested in the edge-creation process (more dialogue, more connections, more proof work). Tools also create their own edges — a dashboard connects metrics to decisions, a workflow engine connects research notes to proofs to papers.

The tools followed the same evolutionary pattern as the research:

1. Pain-driven micro-tools: "my system crashes, I need a monitor"
2. Tools for the AI: enabling autonomous operation, which multiplied output
3. Self-extending system: the AI builds tools when gaps are identified

Each phase increased the throughput of the entire system, freeing more bandwidth for the creative work that generates edges.

4. Local fixing ("if there's a small problem, I solve it")

A shift from "branch to the next interesting thing" (adding disconnected nodes) to "solve the problem in front of me" (adding edges between existing nodes). This is the behavioral change that directly drives edge density upward. Branching creates breadth; local fixing creates depth. Percolation requires depth.

5. Queue drainage

The most fundamental accelerator — and the one that unlocked the other four.

Ideas that are conceived but not realized remain as open cognitive loops (the Zeigarnik effect), consuming working memory bandwidth. A backlog of unrealized ideas creates backpressure on ideation: the mind cannot generate new ideas because it is full of old ones that won't let go. The mechanism is analogous to a **CPU pipeline stall**: when one pipeline stage blocks (waiting for a resource), all upstream stages freeze. The processor sits idle even though work is queued. The creative mind has the same architecture:

$$\text{ideation} \rightarrow \text{formulation} \rightarrow \text{realization} \rightarrow \text{integration}$$

When the realization stage is the bottleneck (ideas arrive faster than they can be implemented), a queue forms. The queue itself suppresses upstream stages: ideation stalls because working memory is saturated with unrealized ideas waiting in line.

Queueing theory gives the exact relationship. Little's Law: $L = \lambda W$, where L is the queue length, λ is the idea arrival rate, and W is the average time-to-realize. When W is high (solo work, no AI, no tools), L grows until backpressure suppresses λ. Collapsing W through AI-augmented realization (papers written in days instead of weeks, proofs completed in hours instead of months, tools built in minutes instead of days) drains the queue, frees cognitive bandwidth, and unleashes λ.

Each drained item does double duty: it frees bandwidth (one fewer open loop) AND creates edges (a realized idea connects to code, proofs, papers, and other realized ideas). Queue drainage is simultaneously a cognitive resource operation and a graph densification operation.

This explains a subtle feature of the flow state: the experience of effortlessness is not the absence of *effort* — it is the absence of *backlog*. The cognitive pipeline runs at full throughput with no stalls,

and it turns out that full-throughput processing is what "effortless" feels like from the inside. The mind's true output rate was always high; it was being throttled by unrealized ideas clogging the queue.

Generative offloading: beyond GTD

The cognitive offloading in this system is qualitatively different from conventional productivity methods.

David Allen's Getting Things Done (GTD) externalizes open loops: write down the task, the mind is freed. The list is a **passive buffer** — it stores items, conserving information without creating it:

$$H(\text{after GTD offload}) = H(\text{before offload})$$

The system described here has three layers of offloading, of which only the first corresponds to GTD:

Layer	What's offloaded	Where it goes	What happens
Task layer (GTD)	Todos, open loops	Task pool, written lists	Stored passively. Mind freed.
Knowledge layer	Theories, insights, realizations	Papers, Lean proofs, knowledge surfaces	Formalized and connected. Creates edges in the knowledge graph.
Generative layer	Ideas that need to be *built*	Realized artifacts: working tools, proved theorems, running code	Contact with reality generates NEW ideas that didn't exist before manifestation.

The generative layer is the critical difference. When you actually write the proof, you discover the Mathlib lemma that connects two domains you hadn't linked. When you actually build the tool, you see the pattern that suggests the next tool. When you actually write the paper, the formalization reveals a gap that IS the next paper. These connections didn't exist in the unrealized idea — they emerge through the act of building. The manifestation is not just offloading; it is **generative**:

$$H(\text{after realization}) > H(\text{before offload})$$

Information is *created* in the realization step. This is **autocatalysis**: each realized idea catalyzes the generation of new ideas, which when realized, catalyze more. The system doesn't just drain the queue — it fills it with *better* items while draining it. But because W (time-to-realize) is now so low, the inflow never creates backpressure. The queue stays short, the mind stays free, and the generation rate stays high.

This distinguishes the system from all three conventional approaches: GTD (layer 1 only — passive storage), knowledge management tools like Zettelkasten or Obsidian (layers 1-2 — storage +

linking), and this system (layers 1-3 — storage + linking + AI-augmented realization where the build process is itself a generative function producing new knowledge as a byproduct).

The reinforcement structure

The five accelerators are not independent — they form a positive feedback loop:

$$\text{cross-fertilization} \xrightarrow{\text{creates edges}} \text{denser graph} \xrightarrow{\text{better navigation}} \text{more productive dialogue} \xrightarrow{\text{generates more edges}} \cdots$$

Queue drainage enables cross-fertilization (freed bandwidth). Tools enable queue drainage (faster realization). Local fixing densifies the graph that tools navigate. Session mining prevents the edges from decaying. The system is autocatalytic.

Stigmergy: Intelligence in the Environment

The coordination mechanism underlying the five accelerators has a name in complex systems theory: **stigmergy** (Grassé, 1959) — from the Greek *stigma* (mark) + *ergon* (work). Literally: work stimulated by the trace of previous work.

Termites build complex mounds without central planning, without inter-individual communication, and without any termite understanding the global structure. The mechanism: each termite deposits material, the deposit modifies the local environment, and the modified environment guides the next termite's behavior. The structure accumulates through local actions. The mound IS the plan — no architect required.

The same mechanism operates in an AI-augmented research system:

Stigmergy (termites)	Knowledge system
Mud ball deposited	A tool built, a paper written, a theorem proved
Modified environment attracts further building	Each artifact changes the repository, which changes what the AI suggests, which changes what the human builds next
No central plan	The system architecture precipitated from solved micro-problems, not from design
The mound IS the coordination	The repository IS the plan — its accumulated structure guides all agents
Multiple termites, no communication	Multiple AI agents, post-hoc coordination through shared state

The key insight of stigmergy that separates it from ordinary collaboration: **the intelligence is in the environment, not in the agents.** No single termite is smart. No single AI session has the full picture. But the accumulated state of the substrate — the graph, the tools, the memory surfaces, the formal kernel — contains more structure than any individual agent contributed. The whole exceeds the sum because the substrate accumulates and the accumulation creates gradients.

This connects directly to the percolation story: stigmergic accumulation is the *mechanism* by which the knowledge graph gains edges. Each local action (a solved problem, a written paper, a built tool) modifies the shared substrate. The modification attracts further action in the same neighborhood. Density accumulates locally, then connects globally — the same dynamics as Erdős–Rényi percolation, but driven by stigmergic feedback rather than random edge addition.

Meta as Grade: The Graded Latent Hierarchy

There is a deeper structural observation that connects the percolation phenomenon to the theory of latent representations.

The grade hierarchy

In a graded latent framework, a system's representation decomposes into levels:

Grade	What it represents	Concrete instance
$k = 0$	The objects themselves	Individual papers, proofs, tools, ideas
$k = 1$	The structure of how they connect	The knowledge graph, the orchestration tooling, the workflow engine
$k = 2$	The structure of the structure	The percolation theory — *why* the graph works, *when* it transitions
$k = 3$	The structure of that theory	Recognizing that "meta = grade+1" — the operation itself

"Thinking about the meta" is not a vague philosophical stance. It is a precise operation: constructing the grade $k + 1$ component of the latent representation while operating at grade k:

$$\text{meta}(X) = \pi_{k+1}(\mathcal{L}(X))$$

where $\mathcal{L}(X)$ is the graded latent of X and π_{k+1} is the projection to grade $k + 1$.

Why grade-building accelerates percolation

Each grade provides the **navigation gradient** for the grade below it. Grade 1 (the graph structure, the tools) tells you where to go at grade 0 (which paper to write, which proof to close). Grade 2 (the percolation theory) explains why grade 1 works and predicts when it will break. Without grade 1, grade 0 is a flat landscape with no gradient — you need external planning. With grade 1, the gradient exists intrinsically.

The creator in question was fixated on two things long before the percolation occurred:

1. **AI-human collaboration** — constructing grade 1 infrastructure: how the edge-generation process works, what tools the AI needs, how to make dialogue productive.
2. **Meta-thinking** — constructing grade 2: understanding the structure of the structure, why the collaboration converges, what makes the system self-sustaining.

These were not separate interests. They were the same operation at different grades of the same latent. Both were adding higher-grade components to the knowledge system's representation. And those higher-grade components are precisely what provides the "effortless direction" experience at grade 0: when the grade hierarchy is populated, navigation at every level reduces to gradient-following on the level above.

The AI wrangler as grade-builder

Roughly two years before the transition, the creator encountered a prediction (attributed to early AI industry commentary) that a new role would emerge: the "AI wrangler" — someone who knows AI systems so intimately that working with them becomes a form of skilled partnership, analogous to an animal wrangler's relationship with their subjects.

The creator became exactly this. But the deeper insight is that an effective AI wrangler is not someone who learned prompting tricks. An AI wrangler is someone who **built the grade 1 and grade 2 components of the human-AI system's latent representation**. They understand not just how to use AI (grade 0), but the structure of how human-AI collaboration works (grade 1), and why that structure produces results (grade 2). The wrangling feels effortless because the full grade hierarchy exists — navigation at every level is gradient-following.

The reflexive dimension

The creator has been developing a theory of latent representations — the claim that a sufficient finite-dimensional representation of a system makes all downstream tasks tractable. The spectral representation captures essential structure; the rest is decodable from it.

The percolation model instantiates this theory on the creator's own knowledge system. Once the graph is dense enough that $\lambda_2 > 0$, the spectral representation exists, and the downstream task of research direction becomes trivially decodable from local structure. The creator built a sufficient representation of their own research landscape — and navigation became effortless. The theory applies to itself.

The tooling makes this concrete: the graph navigation commands literally compute spectral properties of the knowledge graph. The bridge suggestions find edges that would most increase λ_2. The formal system and the cognitive system are isomorphic — the same phase transition, visible from both inside (the felt experience of effortless direction) and outside (the measurable graph density crossing the percolation threshold).

The Language Dimension: Hungarian as Latent, English as Decoded

A final structural observation concerns the role of natural language in the cross-fertilization process.

Why language structure matters

The weak Sapir-Whorf hypothesis — well-supported by experimental work (Boroditsky, 2001; Levinson, 2003) — holds that language does not determine thought but shapes *habitual patterns* of thought. If the creator's native language has structural features that align with the operations required for knowledge-graph construction, those habits become an accelerator.

Hungarian (a Uralic, agglutinative language) has three features that are directly relevant:

1. **Agglutinative morphology**: meaning is built by stacking suffixes onto root words. A single Hungarian word can encode what English requires a full phrase for. Hungarian speakers habitually think in **compressed representations** — more semantic content per token. This is a natural affinity for latent representations: the language's morphology trains its speakers to compress.

2. **18 grammatical cases**: relationships between concepts are encoded morphologically (*-ban, -nak, -ról, -hoz, -ért, -val*), not by word order. Each suffix marks a specific relational role. This means the language natively encodes **edges**, not just nodes. A Hungarian speaker does not just name an object — they name its relationship to other objects. The language's case system is inherently graph-structured.

3. **Obligatory focus position**: Hungarian has relatively free word order, but the position immediately before the verb marks the **focus** — the most important new information in the sentence. Every utterance requires an explicit choice about what matters most. This is a gradient-computation mechanism embedded in syntax: every sentence implicitly ranks information by salience.

A language that trains its speakers to compress, relate, and focus may be pre-adapted for building dense knowledge graphs with clear gradients — though the magnitude of this effect is unknown.

Bilingual dialogue as productive friction

In the system described here, the creator thinks and speaks in Hungarian; the AI responds in English. This cross-lingual asymmetry creates a specific dynamic:

Forced representation change. The creator's Hungarian thought is compressed and relationally encoded. The AI's English response is analytic and positionally structured. Processing the response requires *recoding* from one representational format to another. The specific mechanism: cross-lingual recoding from an analytic language to a case-rich language forces implicit relationships to become explicit (Hungarian case suffixes require choosing a relational role that English leaves vague), and explicit relationships activate more neighboring concepts in the graph. This recoding is an edge-creation operation — but one that only functions when the translation is near-automatic. For a non-fluent bilingual, the overhead would exceed the benefit.

Dual encoding. Every concept is held in two formats: the native Hungarian representation and the English form received from the AI. Dual coding theory (Paivio, 1986) predicts that information encoded in two systems is more richly connected and more easily retrieved.

Hungarian as latent, English as decoded. The structural parallel to the latent representation framework is suggestive: the creator carries the compressed representation (Hungarian thought), the AI provides the decoded, explicit form (English prose). The conversation oscillates between latent and decoded space — the *kibontjuk* dynamic from §5.1.

Honest assessment

The language dimension is a hypothesis, not a demonstrated mechanism. The bilingual cognition literature supports the general claim that cross-linguistic transfer activates broader conceptual networks (Kroll & de Groot, 2005). Hungarian's case system does force relational encoding that English leaves implicit. The combination *should* create additional edges per conversational round.

But the quantitative significance is unknown. The dominant accelerators in this system are cross-fertilization, queue drainage, and tool-building — each of which operates regardless of the languages involved. The language effect is at most a multiplier on the cross-fertilization process, not the main engine. Whether it contributes a 5% or 50% boost to edge-creation rate is an open empirical question — testable by comparing speakers of typologically distant vs. similar language pairs in equivalent AI-collaborative setups.

The Insight

The sustained creative flow state — weeks of productive work without planning, with reliable intuition about what to do next — is not a psychological state that some people achieve through discipline or motivation. It is a **topological property of a knowledge graph** that emerges when the edge density crosses the percolation threshold.

Below the threshold, planning is not a preference but a *structural necessity*: the graph is fragmented, and you cannot navigate between clusters by following local connections. Above the threshold, planning is not just unnecessary but *redundant*: the giant component's topology provides direction at every point.

Five consequences:

1. **The transition is achievable, not innate.** It depends on graph density, not personality. Anyone who accumulates enough *connected* knowledge in a domain (or across domains) will experience it. The key verb is "connected" — isolated nodes (unimplemented ideas, unlinked papers) do not contribute. Only edges count.

2. **Human-AI collaboration is an edge-generation accelerator.** The synergistic information in human-AI dialogue creates cross-domain connections at 10-100× the solo rate. This compresses the time-to-percolation from years to weeks. The AI is not a productivity tool — it is a **phase transition accelerator**.

3. **The flow state is self-reinforcing and irreversible.** Once the giant component exists, it does not fragment under normal conditions (edges are persistent). And the state itself generates more edges (productive work creates new connections), pushing the graph further above threshold. The transition is a one-way door.

4. **Stigmergy replaces architecture.** The system does not need to be designed. It needs to be *used*. Each local action (solve a problem, build a tool, write a paper) modifies the shared substrate, and the accumulated substrate guides the next action. The intelligence is in the environment, not in any individual agent. The plan is the accumulated structure itself.

5. **"Meta" is a computable operation, not a philosophical stance.** Thinking about the meta is constructing the grade $k + 1$ component of the latent representation. Each

grade provides the navigation gradient for the grade below. When the full grade hierarchy is populated — objects, their structure, the theory of that structure — navigation at every level becomes gradient-following. The apparent mystery of "knowing without thinking" dissolves: the higher grades contain the gradients that the lower grades follow.

The planning-versus-flow debate in productivity literature is a false dichotomy. It is not a question of preference or personality. It is a question of *graph density* and *grade completeness*. Below threshold: plan. Above threshold: follow the gradient. The only question is how fast you can add edges — and how high you can build the grade hierarchy that makes those edges navigable.

References

- Erdős, P. and Rényi, A (1960). On the evolution of random graphs. *Publications of the Mathematical Institute of the Hungarian Academy of Sciences*, 17-61.
- Csíkszentmihályi, M (1990). Flow: The Psychology of Optimal Experience. *Flow: The Psychology of Optimal Experience*.
- Fiedler, M (1973). Algebraic connectivity of graphs. *Czechoslovak Mathematical Journal*, 23(2), 298-305.
- Zeigarnik, B (1927). Über das Behalten von erledigten und unerledigten Handlungen. *Psychologische Forschung*, 1-85.
- Little, J.D.C (1961). A proof for the queuing formula: $L = \lambda W$. *Operations Research*, 9(3), 383-387.
- Dreyfus, S.E. and Dreyfus, H.L (1986). Mind over Machine: The Power of Human Intuition and Expertise in the Era of the Computer. *Mind over Machine: The Power of Human Intuition and Expertise in the Era of the Computer*.
- Kahneman, D (2011). Thinking, Fast and Slow. *Thinking, Fast and Slow*.
- Grassé, P.-P (1959). La reconstruction du nid et les coordinations interindividuelles chez *Bellicositermes natalensis* et *Cubitermes* sp. La théorie de la stigmergie. *Bellicositermes natalensis*, 6(1), 41-80.
- Allen, D (2001). Getting Things Done: The Art of Stress-Free Productivity. *Getting Things Done: The Art of Stress-Free Productivity*.
- Boroditsky, L (2001). Does language shape thought? Mandarin and English speakers' conceptions of time. *Cognitive Psychology*, 43(1), 1-22.
- Levinson, S.C (2003). Space in Language and Cognition: Explorations in Cognitive Diversity. *Space in Language and Cognition: Explorations in Cognitive Diversity*.
- Paivio, A (1986). Mental Representations: A Dual Coding Approach. *Mental Representations: A Dual Coding Approach*.