

Systems Resilience and Autonomous Recovery Architectures in Decentralized Computational Colonies

The operational stability of a decentralized colony within the Screeps environment is predicated on the continuous cycle of energy extraction, transportation, and utilization. When this cycle is disrupted—a state colloquially referred to as a "colony blackout" or a "crippled system"—the colony loses its ability to sustain its specialized labor force. In such scenarios, the primary challenge is not merely the restoration of power, but the minimization of the recovery window to prevent the loss of critical infrastructure, such as the Room Controller Level (RCL) or defensive fortifications.¹ This analysis examines the mathematical and algorithmic frameworks employed by elite practitioners to facilitate rapid bootstrapping, with a specific focus on body part optimization, hatchery energy prioritization, and stimulus-response logic.

Theoretical Foundations of Emergency Energy Dynamics

The fundamental safety mechanism provided by the game engine is the passive regeneration of the spawn structure. A spawn will automatically accumulate one energy unit per tick if the total energy stored in all spawns and extensions within the room is below 300 units.¹ This mechanism ensures that a player is never permanently excluded from the game world, provided their Room Controller remains intact. However, relying solely on this passive rate is an inefficient strategy that top-tier players seek to bypass through "bootstrapping" algorithms.

The passive regeneration rate (1 e/t) is significantly lower than the potential output of a single energy source, which regenerates 3,000 units every 300 ticks, equating to 10 e/t .⁴ Consequently, every tick spent waiting for the spawn to passively regenerate represents a 90% loss in potential energy throughput. High-performance AI architectures prioritize the transition from passive regeneration to active harvesting at the earliest possible threshold.⁷

Mathematical Optimization of Emergency Creep Morphology

The central question in recovery logic is whether to spawn a minimalist creep at the earliest possible energy threshold or to wait for a more capable unit. The user's hypothesis suggests that a two-part creep (e.g., ``) might be superior to waiting for a full 300-energy buffer. To evaluate this, one must analyze the physical weight, build cost, and operational limitations of various configurations.

Creep Template	Body Composition	Cost (Energy)	Ticks to Regenerate (Passive)	Throughput (Active)	Operational Role
Minimalist Miner	``	150	150	2 e/t	Drop-harvesting only; cannot transport.
Emergency Hauler	``	100	100	N/A	Transport only; cannot harvest.
Standard Pioneer	``	200	200	2 e/t	Full cycle (Harvest, Transport, Deliver).
Specialized Worker	``	250	250	4 e/t	High-speed harvesting; requires hauler.
Reinforced Pioneer	``	300	300	4 e/t	Efficient recovery; limited capacity.

Data indicates that the build cost of a creep is the sum of its parts: WORK costs 100, MOVE costs 50, and CARRY costs 50.⁸ A creep (150 energy) can begin harvesting at $T=150$. However, without a `CARRY` part, it cannot hold energy or transfer it to a spawn or extension.⁸ It can only drop-harvest onto the ground or into a container.⁴ In a total blackout where no haulers exist, a creep is effectively useless for refilling the spawn unless a container already exists beneath it or a secondary hauler is spawned immediately after.⁶

Conversely, the `` Pioneer (200 energy) represents the "Goldilocks" threshold for recovery. While it takes 50 ticks longer to spawn than the minimalist miner, its ability to carry 50 units of energy allows it to complete the delivery loop autonomously.⁷ By $T = 235$ (accounting for

movement and harvest time), a Pioneer can have delivered its first 50 units back to the spawn, effectively outperforming the passive regeneration rate by ^{500%} once it is active.

The Impact of Fatigue and Terrain on Recovery Speed

Movement efficiency is a critical variable in bootstrapping. Fatigue is generated by non-MOVE parts and reduced by MOVE parts. The formula for movement speed ^{*t*} (ticks per move) is defined as $t = \lceil (K \times W) / M \rceil$, where *W* is the number of non-MOVE parts (excluding empty CARRY), *K* is the terrain factor, and *M* is the number of MOVE parts.⁸

Configurati on	Terrain	Weight (W)	MOVE (M)	Fatigue (F)	Ticks per Tile
``	Plain	1	1	0	1
``	Swamp	1	1	8	5
`` (Full)	Plain	2	1	2	2
`` (Empty)	Plain	1	1	0	1

For an emergency Pioneer, the empty return trip to the source is conducted at maximum speed (1 tile/tick), while the return trip with energy is slowed (2 tiles/tick on plains).⁸ Top-tier algorithms calculate the "Time-to-Hatchery-Full" by integrating the distance to the source, the harvesting rate of the WORK parts, and the fatigue-induced travel delay.⁷ If the source is more than 25 tiles away, the "cost" of the move parts and the time lost in transit may justify spawning a minimalist miner and a separate hauler rather than a multi-role pioneer.¹⁰

Algorithmic Architectures for Autonomous Recovery

The most resilient colonies do not treat recovery as a special case but as a high-priority state within a unified task management system. Expert AI systems, such as the "Overmind" architecture, utilize a hierarchical process model involving Directives, Overlords, and the Overseer.¹⁴

The Role of the Overseer in Colony Crash Detection

The "Overseer" serves as the primary stimulus-response engine.¹⁴ Each tick, it examines room conditions for "anomalous states," specifically targeting energy deficits and population

collapses. A "Colony Crash" is identified when the following conditions are met:

1. The population of energy-extracting and energy-transporting creeps is zero.
2. The room's available energy is below the threshold for the standard operating creep (e.g., 1,200 energy).
3. The room possesses an active spawn or the materials to build one.

Upon detection, the Overseer places a "Bootstrapping Directive" on the room.¹⁴ This directive acts as a signal to the spawning system to ignore all secondary tasks (upgrading, building, or defending non-vital structures) and focus exclusively on producing "Bootstrapper" units.¹⁴

Bootstrapping Overlords and Spawning Priority

The logic for managing these emergency units is contained within a "Bootstrapping Overlord".¹⁴ This object handles the request and control of Pioneers. In a high-tier system, spawning requests are sorted by priority. While a standard harvester might have a priority of 5, a Bootstrapper is assigned a priority of 0 (the highest possible), ensuring that as soon as the spawn hits the 200-energy threshold, the Pioneer is produced.¹⁴

This system bypasses the "hatchery waiting" problem mentioned in the user's query. A naive algorithm might check if `(room.energyAvailable == room.energyCapacityAvailable)`, which at RCL 8 would mean waiting for 12,900 energy—an impossible task during a blackout.² The Bootstrapping Overlord instead calculates the Minimum Viable Creep (MVC) cost and issues the `spawnCreep` command based on the immediate `room.energyAvailable`.¹²

Task Parentage and State Persistence

To ensure efficiency, recovery units utilize "Task Parentage".¹⁴ When a Bootstrapper finishes refilling the spawn (its primary task), it does not idle. Its "parent task" is typically "Harvest from Source," which it resumes automatically. This reduces CPU overhead and ensures that the creep is always moving toward a goal.¹⁴

Hatchery Logistics and Extension Refilling Strategies

Once the initial recovery creeps are active, the bottleneck shifts from harvesting to refilling the extension network. At higher RCLs, the number of extensions can reach 60, and their layout significantly impacts the speed of recovery.²

Optimal Refilling Algorithms

Top players avoid the use of `findClosestByPath` during recovery because it is CPU-intensive and can lead to inefficient "ping-ponging" between targets.¹⁹ Instead, they implement "Static

Routing" or "Flood-Filling" algorithms.²⁰

Algorithm Type	Mechanism	Efficiency	Use Case
Naive Search	Calls findClosestByPath every tick.	Low (High CPU)	Initial RCL 1 setup.
Memoized List	Stores extension IDs in a list; iterates.	Medium	RCL 2-4 with few extensions.
Static Optimal Path	Pre-calculates a Hamiltonian path through all extensions.	High	High RCL (6-8) "Bunker" layouts.
Flood-Fill	Creep fills every adjacent extension in a single pass.	Very High	Specialized "Refiller" creeps in tight layouts.

In the "Overmind" Hatchery HiveCluster, a dedicated "Refiller" creep often sits in a fixed position adjacent to the spawn and multiple extensions, utilizing links or containers to transfer energy without moving.¹³ During a recovery, however, if these stationary positions are not yet staffed, the Bootstrappers use a sorted ID array to refill the structures in a deterministic order.²⁰ This ensures that the spawner is refilled first, enabling the production of more creeps even before the extensions are fully powered.

Body Part Scaling During Reconstruction

As energy levels rise, the algorithm should transition from the 200-energy Pioneer to specialized units. A common mistake is scaling body parts too quickly. If an AI spawns a 3,000-energy creep as soon as the room hits that capacity, it risks another blackout if that expensive creep dies unexpectedly.³

The "Spawn Time Governor" algorithm tracks the "spawner saturation"—the percentage of time the spawner is active.²² During recovery, the goal is 100% saturation with small creeps. As the "Sink" (energy usage) balances with "Source" (energy input), the algorithm gradually increases the creep size.⁷

The Role of Static Infrastructure in Systems Resilience

Colonies that recover the fastest are those that invest in "Buffer Infrastructure." This includes Containers, Links, and Storage.²

Static Harvesting and Buffering

Top-tier players utilize "Static Harvesting," where a miner stays permanently at the source.⁶

- **Source Containers:** Placing a container under a miner allows it to harvest even when its CARRY capacity is full, as the overflow is captured by the container.⁴ During a recovery, a Pioneer can instantly withdraw 2,000 energy from a source container, bypassing the harvesting delay entirely.³
- **Tombstones and Ruin Recovery:** When creeps die, they leave tombstones. A high-tier recovery algorithm searches for Tombstones and DroppedResources before attempting to harvest from a source, as these represent "pre-processed" energy with zero WORK part requirements.⁴

Link-Based Recovery Networks

At RCL 5, the "Link" structure becomes the primary tool for resilience.² A "Hatchery Link" is maintained at full capacity. If the colony crashes, the first Pioneer only needs to harvest enough energy to fill the "Source Link." The energy is then transmitted instantly to the hatchery, where a second creep (perhaps a minimalist hauler) can distribute it to the extensions.¹⁸

Defensive Posture During Bootstrapping

A crippled colony is highly vulnerable to harassment. A single enemy scout can kill an emergency Pioneer, reset the passive regeneration timer, and potentially force a respawn.¹

The "Safe Mode" Strategic Window

Safe mode is the most powerful tool for recovery, providing 20,000 ticks of absolute protection.¹

- **Automatic Triggering:** Resilience-focused AI monitors the room's "Threat Level." If the energyAvailable is below the recovery threshold and a hostile creep is detected with ATTACK or WORK parts (for dismantling), the AI triggers activateSafeMode immediately.²⁶
- **Safe Mode Conservation:** Because safe mode activations are limited, the AI must distinguish between a minor probe and a "Room Wipe" threat.²⁶

Tower-Based Defense Recovery

In RCL 3+ rooms, a "Tower" is the first structure to be refilled after the spawn.¹⁸ A tower can defend the entire room with zero movement cost, allowing recovery units to work in safety.²⁴ Elite algorithms prioritize "Tower Refilling" over "Extension Refilling" if hostile creeps are present, recognizing that the loss of the bootstrap creep is more costly than a delay in spawning the

second one.²⁴

Mathematical Modeling of the "Fastest Manner" to Recover

To determine the absolute fastest recovery algorithm, we must compare the "Pioneer" strategy against the "Split-Role" strategy in a simulation of a 0-energy room with a source 10 tiles away.

Scenario A: The Pioneer Algorithm

1. Passive Regen to 200 Energy: 200 ticks .
2. Travel to Source: 10 ticks .
3. Harvest 50 Energy: 25 ticks (at 2 e/t).
4. Return to Spawn: 20 ticks (at 2 t/tile due to fatigue).
5. **Total for first 50 Energy delivery: 255 ticks .**

Scenario B: The Split-Role Algorithm (User Hypothesis)

1. Passive Regen to 150 Energy: 150 ticks .
2. Spawn Miner ``: $T = 150$.
3. Passive Regen to 100 Energy: 100 ticks .
4. Spawn Hauler [C, M]: $T = 250$.
5. Miner arrives at source at $T = 160$; begins drop-harvesting.
6. Hauler arrives at source at $T = 260$; picks up 50 energy.
7. Hauler returns to spawn at $T = 270$.
8. **Total for first 50 Energy delivery: 270 ticks .**

The "Pioneer" algorithm is mathematically superior by 15 ticks (approx. 45-60 seconds of real-time).⁷ This advantage compounds with every trip. By the time the split-role hauler returns once, the Pioneer is already halfway through its second trip. Furthermore, the Pioneer strategy is more "CPU-resilient"; if the hauler is killed, the miner cannot contribute to the spawner, whereas the Pioneer can always complete its cycle.

Exception: The "Container" Variable

If a container exists at the source with 2,000 energy, the math changes.

1. Passive Regen to 100 Energy: 100 ticks .
2. Spawn Hauler [C, M]: $T = 100$.
3. Hauler arrives at container: $T = 110$.
4. Hauler returns to spawn: $T = 120$.
5. **Total for first 50 Energy delivery:** 120 ticks .

Therefore, the "ideal" algorithm is a **Conditional Morphology Selector**:

- IF energy exists in a container/storage/tombstone: Spawn a 100-energy Hauler first.
- ELSE IF no energy exists: Spawn a 200-energy Pioneer first.⁷
- NEVER wait for the spawner to hit 300 if a 200-energy creep can do the job.⁷

Advanced Recovery: The "Assimilator" and Inter-Colony Support

For players with a Global Control Level (GCL) higher than 1, the fastest recovery method is often not internal to the room but external.¹

The "Global Hivemind" Approach

In the "Assimilator" model (used by Overmind), a crippled colony issues a "Global Help Request".¹⁵

- **Neighboring Support:** An adjacent, healthy colony detects the crash. It spawns a "Transporter" with 25 CARRY parts and sends it across the room border.²⁵
- **Cross-Room Refilling:** A single large transporter (1,250 energy) can refill an entire RCL 4 hatchery in one trip, bypassing the need for any internal recovery creeps.³
- **Inter-Shard Portals:** In very high-level play, if an entire shard is compromised, players use portals to send recovery forces from Shard 1 to Shard 0.¹⁸

Strategic Redeployment (Respawning)

If a room is under constant siege and bootstrapping is impossible, top-tier players utilize the "Respawn" feature.¹ Because GCL is permanent, a player can restart in a new sector with the ability to instantly claim 3+ rooms, allowing for "parallel development" which is often faster than trying to recover a ruined, contested room.¹

Systems Engineering Recommendations for Autonomous Resilience

The analysis of top-tier strategies suggests a "Multi-Layered Recovery Protocol" that should be encoded into any high-performance AI.

Protocol Layer 1: Detection and Safe-State

The AI must monitor `room.energyAvailable` against `room.energyCapacityAvailable`. If the ratio falls below 10% and population is zero, enter "CRITICAL_BLACKOUT" state. If hostile presence is > 0 , activate Safe Mode.²⁶

Protocol Layer 2: Morphology Optimization

The AI must not use a static "Harvester" template. It should use a dynamic builder that selects the smallest possible body that can fulfill the "Harvest-Transport" loop.¹⁰ This is typically `` for 200 energy.

Protocol Layer 3: Hatchery Prioritization

Modify the energy distribution logic. Standard logic often fills Towers or Labs first. Recovery logic must prioritize `STRUCTURE_SPAWN` and then `STRUCTURE_EXTENSION` to maximize the spawner's potential as quickly as possible.²⁰

Protocol Layer 4: Traffic Management

Implement "Active Shoving." Emergency creeps must have the right-of-way. Any creep not currently carrying energy or harvesting should be moved out of the "Source-to-Spawn" path to prevent congestion.³²

By adhering to these principles, a Screenshot colony transitions from a fragile entity to an "anti-fragile" system, capable of self-correcting from a zero-population state in the minimum number of game ticks allowed by the engine's physics. The integration of mathematical morphology, static routing, and inter-colony support forms the triad of modern Screenshot resilience.

Works cited

1. Respawning - Screenshot Documentation, accessed February 21, 2026, <https://docs.screenshot.com/respawn.html>
2. Control | Screenshot Documentation, accessed February 21, 2026, <https://docs.screenshot.com/control.html>
3. How do I build out of my spawn's energy? :: Screenshot: World General Discussions, accessed February 21, 2026, <https://steamcommunity.com/app/464350/discussions/0/3223871682609932118/>

4. Energy - Screeps Wiki, accessed February 21, 2026, <https://wiki.screepspl.us/Energy/>
5. Efficiency | Screeps Forum, accessed February 21, 2026, <https://screeps.com/forum/topic/1195/efficiency>
6. What's the best strategy for gathering energy? - screeps - Reddit, accessed February 21, 2026, https://www.reddit.com/r/screeps/comments/4x8z65/whats_the_best_strategy_for_gathering_energy/
7. Screeps #4: Pipeline Optimization | Field Journal, accessed February 21, 2026, <https://jonwinsley.com/notes/screeps-pipeline-optimization>
8. Creep | Screeps Wiki | Fandom, accessed February 21, 2026, <https://screeps.fandom.com/wiki/Creep>
9. Screeps Arena Documentation, accessed February 21, 2026, <https://arena.screeps.com/docs/>
10. Questions on roles | Screeps Forum, accessed February 21, 2026, <https://screeps.com/forum/topic/913/questions-on-roles>
11. Creeps - Screeps Documentation, accessed February 21, 2026, <https://docs.screeps.com/creeps.html>
12. Creep Body Setup Strategies - Screeps Wiki, accessed February 21, 2026, https://wiki.screepspl.us/Creep_body_setup_strategies/
13. Is there a method to fully utilize the energy resource which is only available to one creep at a time? : r/screeps - Reddit, accessed February 21, 2026, https://www.reddit.com/r/screeps/comments/eqw2gt/is_there_a_method_to_fully_utilize_the_energy/
14. Screeps #1: Overlord overload - Ben Bartlett, accessed February 21, 2026, <https://bencbartlett.com/blog/screeps-1-overlord-overload/>
15. bencbartlett/Overmind: AI for Screeps, a multiplayer programming strategy game - GitHub, accessed February 21, 2026, <https://github.com/bencbartlett/Overmind>
16. Screeps #0: A Brief History of Game.time | Ben Bartlett, accessed February 21, 2026, <https://bencbartlett.com/blog/screeps-0-a-brief-history-of-game-time/>
17. New spawn extension system | Screeps Blog, accessed February 21, 2026, <https://blog.screeps.com/2015/05/New-spawn-extension-system/>
18. Getting Started - Screeps Wiki, accessed February 21, 2026, https://wiki.screepspl.us/Getting_Started/
19. Common Development Problems - Screeps Wiki, accessed February 21, 2026, https://wiki.screepspl.us/Common_development_problems/
20. How can i store all my extensions/spawns in order? | Screeps Forum, accessed February 21, 2026, <https://screeps.com/forum/topic/2426/how-can-i-store-all-my-extensions-spawns-in-order>
21. First attempt at logic to automate creep creation | Screeps Forum, accessed February 21, 2026, <https://screeps.com/forum/topic/1484/first-attempt-at-logic-to-automate-creep-creation>
22. ryanrolds/screeps-bot-choreographer: A bot for Screeps - GitHub, accessed

- February 21, 2026, <https://github.com/ryanrolds/screeps-bot-choreographer>
23. Is there a guide out there for the ideal ratios of your creeps? : r/screeps - Reddit, accessed February 21, 2026,
https://www.reddit.com/r/screeps/comments/864qy3/is_there_a_guide_out_there_for_the_ideal_ratios/
 24. Offensive Strategy : r/screeps - Reddit, accessed February 21, 2026,
https://www.reddit.com/r/screeps/comments/7d6w2t/offensive_strategy/
 25. Creeps in wrong room : r/screeps - Reddit, accessed February 21, 2026,
https://www.reddit.com/r/screeps/comments/6xdv84/creeps_in_wrong_room/
 26. Controller | Screeps Wiki - Fandom, accessed February 21, 2026,
<https://screeps.fandom.com/wiki/Controller>
 27. PvP Game discussion | Screeps Forum, accessed February 21, 2026,
<https://screeps.com/forum/topic/473/pvp-game-discussion/28>
 28. Best Attack Strategy? | Screeps Forum, accessed February 21, 2026,
<https://screeps.com/forum/topic/2995/best-attack-strategy>
 29. Damage order of operations with partially damaged boosted tough and pre-healing | Screeps Forum, accessed February 21, 2026,
<https://screeps.com/forum/topic/2801/damage-order-of-operations-with-partially-damaged-boosted-tough-and-pre-healing>
 30. Creep keeps switching between 2 rooms | Screeps Forum, accessed February 21, 2026,
<https://screeps.com/forum/topic/967/creep-keeps-switching-between-2-rooms>
 31. Game restart - forum - Screeps, accessed February 21, 2026,
<https://screeps.com/forum/topic/947/game-restart>
 32. Screeps #25: Arena - Grouping Up - Field Journal, accessed February 21, 2026,
<https://jonwinsley.com/notes/screeps-arena-grouping-up>