# Patent Claims Draft: Memory-Optimized RL for Metabolic Engineering

## Independent Claims

### Claim 1

A computer-implemented method for optimizing a cellular redox balance comprising:

1. importing a genome-scale metabolic network comprising at least 400 reactions into a memory of a computing device;
2. executing a reinforcement-learning agent that iteratively modifies the network by introducing one or more enzyme sink reactions, each modification comprising a reaction identifier, cellular compartment designation, and expression level parameter;
3. implementing memory optimization during said executing step by:
   1. maintaining a single working copy of the metabolic network across multiple training episodes to eliminate repeated memory allocation,
   2. limiting a cache of flux balance analysis solutions to a predetermined maximum size,
   3. converting solution objects containing pandas DataFrames to minimal data structures,
   4. performing forced garbage collection at predetermined intervals;
4. computing, for each iteration, an objective function that maximizes NAD+ regeneration subject to a biomass-growth penalty constraint;
5. training the reinforcement-learning agent using a distributed IMPALA architecture with at least 32 parallel actor processes;
6. outputting a ranked list of enzyme constructs achieving an improvement in the NAD+/NADH ratio of at least 10% relative to a baseline;
7. storing the ranked list in a non-transitory computer-readable medium.

### Claim 2

A computer-implemented system for memory-efficient reinforcement learning on biological networks comprising:

1. a distributed computing cluster comprising a plurality of actor processes, each actor process configured to:
   1. maintain a working copy of a metabolic model without performing deep copy operations during environment resets,
   2. cache flux balance analysis results in a size-limited data structure,
   3. monitor memory usage and trigger garbage collection when memory exceeds a predetermined threshold;
2. a learner process configured to update neural network parameters based on experiences collected from the actor processes;
3. a memory monitoring system configured to:
   1. track memory usage of each actor process,
   2. restart actor processes that exceed memory limits,
   3. log memory statistics for performance analysis;
4. a checkpoint management system configured to periodically save model states to cloud storage;
5. wherein the system maintains stable memory usage over training periods exceeding 4 hours without memory-related failures.

### Claim 3

A computer-implemented method for designing enzyme constructs for redox optimization comprising:

1. providing a dual-agent reinforcement learning framework comprising:
   1. a first agent representing a biological system optimizing for metabolic efficiency,
   2. a second agent designing enzyme modifications to improve redox balance;
2. defining an action space for the second agent comprising:
   1. selection from a library of redox-active enzymes,
   2. specification of subcellular compartment targeting,
   3. determination of enzyme expression levels;
3. training both agents through competitive optimization where the first agent challenges enzyme designs created by the second agent;
4. implementing memory optimization techniques that reduce memory usage by at least 80% compared to naive implementations;
5. generating enzyme construct recommendations that maintain at least 95% of baseline cellular growth rate while improving redox metrics.

## Dependent Claims

### Claim 4

The method of claim 1, wherein step (b) employs an IMPALA actor-critic architecture with between 32 and 120 parallel actors.

### Claim 5

The method of claim 1, wherein the memory optimization in step (c)(ii) implements a first-in-first-out (FIFO) cache eviction policy when the cache exceeds 5000 entries.

### Claim 6

The method of claim 1, wherein the metabolic network is derived from a human genome-scale metabolic model comprising at least 10,000 reactions.

### Claim 7

The method of claim 1, wherein the enzyme sink reactions are selected from a library comprising NADH oxidases, malate-aspartate shuttle components, and redox-active transporters.

### Claim 8

The method of claim 1, wherein the cellular compartment designation comprises selection from cytosol, mitochondria, and peroxisome compartments.

### Claim 9

The method of claim 1, wherein the reinforcement learning agent is trained on cloud computing infrastructure comprising at least 1TB of total system memory.

### Claim 10

The method of claim 1, wherein the objective function further comprises a penalty term for activation of hypoxia-inducible factor (HIF) pathways.

### Claim 11

The method of claim 2, wherein each actor process is allocated a maximum memory limit of 14GB and is automatically restarted if this limit is exceeded.

### Claim 12

The method of claim 2, wherein the checkpoint management system uploads model states to Amazon S3 storage at intervals of 30 seconds or less.

### Claim 13

The method of claim 2, wherein the memory monitoring system generates alerts when total system memory usage exceeds 80% of available capacity.

### Claim 14

The method of claim 3, wherein the library of redox-active enzymes comprises at least: (a) NADH oxidase from Escherichia coli (NOX\_Ec), (b) NADH oxidase from Lactobacillus brevis (NOX\_Lb), (c) mitochondrial aspartate aminotransferase (mAspAT), (d) cytosolic aspartate aminotransferase (cAspAT), (e) malate dehydrogenase isoforms (MDH1, MDH2).

### Claim 15

The method of claim 3, wherein the competitive optimization employs a reward function that penalizes the first agent for reduced growth rate and rewards the second agent for improved redox balance.

### Claim 16

The method of claim 1, further comprising validating designed enzyme constructs through flux balance analysis simulation on a full genome-scale metabolic model.

### Claim 17

The method of claim 1, wherein the improvement in NAD+/NADH ratio is measured as increased flux through NADH-consuming sink reactions.

### Claim 18

The method of claim 2, wherein the distributed computing cluster operates on Amazon Web Services EC2 instances of type r7i.48xlarge or equivalent.

### Claim 19

The method of claim 1, wherein step (c)(iii) converts COBRApy Solution objects to dictionaries containing only essential flux values as numpy arrays.

### Claim 20

The method of claim 1, wherein the method successfully completes at least 10,000,000 training steps without memory-related system failures.

## Method Claims Focused on Specific Innovations

### Claim 21

A computer-implemented method for preventing memory leaks in biological simulation software comprising:

1. identifying that repeated calls to model.copy() operations in COBRApy consume excessive memory during reinforcement learning training;
2. implementing a scratch model reuse system that maintains a single working copy of a metabolic model across multiple simulation episodes;
3. replacing full model copying with targeted state reset operations that restore the working model to initial conditions;
4. thereby reducing memory allocation from repeated model copying by at least 90%.

### Claim 22

A computer-implemented method for managing flux balance analysis result caching comprising:

1. implementing a size-limited cache for storing solutions to linear programming problems in metabolic simulations;
2. monitoring cache size and implementing first-in-first-out eviction when the cache exceeds a predetermined threshold;
3. maintaining cache hit rates above 70% while preventing unbounded memory growth;
4. wherein the predetermined threshold is between 1000 and 10000 cache entries.

### Claim 23

A computer-implemented method for cloud-based metabolic engineering comprising:

1. deploying a distributed reinforcement learning system across multiple cloud computing instances;
2. implementing automatic checkpointing of neural network states to cloud storage at regular intervals;
3. providing memory monitoring and automatic restart capabilities for distributed processes;
4. enabling fault-tolerant training that can resume from saved states after system failures;
5. wherein the system maintains stable operation for training periods exceeding 4 hours.

## Apparatus Claims

### Claim 24

A computer system for memory-efficient biological network optimization comprising:

1. a processor configured to execute reinforcement learning algorithms;
2. memory comprising at least 32GB RAM for storing metabolic network models;
3. storage comprising solid-state drives for checkpoint data and results;
4. network interface for cloud storage connectivity;
5. software modules comprising:
   1. a metabolic model management module implementing memory-efficient model handling,
   2. a reinforcement learning training module implementing IMPALA architecture,
   3. a memory monitoring module tracking resource usage,
   4. a checkpoint management module providing fault tolerance;
6. wherein the system is configured to perform the method of any of claims 1-23.

### Claim 25

A non-transitory computer-readable medium containing instructions that, when executed by a processor, cause the processor to perform the method of any of claims 1-23.

## Broad Coverage Claims

### Claim 26

A computer-implemented method for artificial intelligence-guided metabolic engineering comprising optimizing enzyme selection and placement using machine learning algorithms trained on genome-scale metabolic models, wherein memory usage is maintained below predetermined limits through systematic optimization of data structures and caching strategies.

### Claim 27

A computer-implemented system for biological network optimization using distributed reinforcement learning, wherein the system implements memory management techniques that enable stable training on models comprising more than 1000 biochemical reactions.

### Claim 28

A computer-implemented method for designing biological systems comprising using competitive multi-agent reinforcement learning to optimize metabolic pathway modifications while maintaining system viability constraints.

**Total Claims**: 28  
**Independent Claims**: 3  
**Dependent Claims**: 25  
**Claim Categories**: Method (18), System (4), Apparatus (2), Medium (1), Broad (3)

**Patent Strategy Notes**: - Claims 1-3 provide broad independent coverage of key innovations - Claims 4-20 provide detailed dependent protection - Claims 21-23 focus on specific technical solutions - Claims 24-25 provide apparatus and medium coverage  
- Claims 26-28 provide very broad fallback positions

**Filing Recommendation**: Include all claims in provisional application to preserve maximum scope for future non-provisional filing and claim refinement.