

Berdasarkan sumber referensi yang Anda berikan, struktur fungsi reward linear menyeimbangkan *throughput* dan *latensi* melalui mekanisme **Penjumlahan Terbobot (Weighted Sum)** yang dinormalisasi.

Dalam konteks *Deep Reinforcement Learning* (DRL) untuk jaringan, agen tidak bisa memaksimalkan *throughput* dan meminimalkan *latensi* secara bersamaan tanpa kompromi, karena kedua metrik ini seringkali bertentangan (misalnya, antrian panjang meningkatkan *throughput* tetapi memperburuk *latensi*).

Berikut adalah detail struktur fungsi reward linear yang direkomendasikan berdasarkan literatur:

1. Struktur Dasar: Penjumlahan Terbobot (Weighted Sum)

Bentuk paling umum dari fungsi reward linear adalah menjumlahkan metrik performa yang diinginkan (diberi tanda positif) dan mengurangi metrik yang tidak diinginkan (diberi tanda negatif).

Formula dasarnya adalah 1, 2, 3:

$$R_t = \alpha \cdot \text{Throughput}_t - \beta \cdot \text{Latency}_t - \gamma \cdot \text{Loss}_t$$

- Throughput_t : Kecepatan data rata-rata (misal: Mbps atau kbps).
- Latency_t : Penundaan rata-rata (misal: ms atau detik).
- α, β, γ : Koefisien (bobot) yang menentukan prioritas agen.

Contoh Penerapan: Dalam sebuah studi optimasi aliran trafik, peneliti menetapkan $\alpha=1$ (untuk throughput), $\beta=0.5$ (untuk delay), dan $\gamma=0.2$ (untuk packet loss) untuk memprioritaskan throughput namun tetap memberikan penalti moderat pada delay.

2. Normalisasi Skala (Scale Normalization)

Masalah utama fungsi linear murni adalah perbedaan unit (Mbps vs milidetik). Jika *throughput* bernilai 500 (Mbps) dan *latensi* bernilai 0.01 (detik), agen akan mengabaikan latensi karena nilainya terlalu kecil secara numerik.

Oleh karena itu, struktur reward harus dinormalisasi, biasanya menggunakan nilai maksimum historis atau kapasitas *link* 4, 5:

$$R_t = w_1 \cdot \frac{T_t}{T_{\max}} - w_2 \cdot \frac{D_t}{D_{\min}}$$

Atau menggunakan skoring berbasis rasio target seperti pada algoritma **Aurora/Genet** untuk *Congestion Control* 1, 6:

- Throughput diukur dalam kbps.
- Latency diukur dalam detik.
- Loss diukur dalam persentase.
- Bobot disesuaikan secara drastis (misal: bobot Latency = -1000) untuk mengimbangi skala unit yang kecil.

3. Pendekatan Berbasis "Regret" (Untuk SLA Violation)

Untuk kasus Anda yang memiliki **SLA ketat** (Port 4 Healthcare), fungsi linear biasa mungkin tidak cukup tegas. Referensi menyarankan penggunaan **Regret-Based Reward** yang linier.

Alih-alih hanya mengurangi latensi, Anda menghitung "penyesalan" (seberapa jauh melanggar target). Berdasarkan **Zeng (2025)**, struktur ini diformulasikan sebagai 7, 8, 9:

$$R_{\text{total}} = -(\lambda_p \cdot r_p + \lambda_d \cdot r_d)$$

Dimana komponen penalti (r) dihitung hanya jika melanggar target:

- **Throughput Regret (r_p):** $\max(\frac{\text{Target} - \text{Actual}}{\text{Target}}, 0)$
- **Latency Regret (r_d):** $\max(\frac{\text{Actual} - \text{Target}}{\text{Target}}, 0)$

Struktur ini memungkinkan Anda menetapkan **prioritas berbeda antar slice**. Misalnya, untuk slice Healthcare, bobot λ_d (latency) dibuat sangat besar, sedangkan untuk slice Kamera, bobot λ_p (throughput) yang diperbesar 9, 10.

4. Reward Shaping dengan Penalti Tambahan

Untuk mempercepat konvergensi dan memastikan keamanan (safety), struktur linear sering ditambahkan dengan komponen penalti non-linear atau *step-function* jika kondisi menjadi "tidak aman" (misal: latensi > ambang batas kritis).

- **Hard Penalty:** Jika latensi > batas SLA, berikan nilai negatif besar konstan (misal -100) 11.
- **Soft Penalty:** Menggunakan fungsi eksponensial atau sigmoid sebagai pengali penalti saat mendekati batas pelanggaran 12, 13.

Kesimpulan untuk Desain Paper Anda

Menggabungkan referensi di atas, berikut adalah struktur reward linear yang optimal untuk skenario 3 Port Anda:

$$R = \underbrace{w_1 \cdot \frac{T_{\text{Port2}}}{T_{\text{Target}}}}_{\text{Insentif Kamera}} - \underbrace{w_2 \cdot \frac{D_{\text{Port4}}}{D_{\text{SLA}}}}_{\text{Penalti Latency}} - \underbrace{w_3 \cdot I(D_{\text{Port4}} > \text{SLA})}_{\text{Pelanggaran Berat}}$$

- Gunakan normalisasi agar T dan D sebanding 4.
- Gunakan bobot w_1, w_2 untuk menyeimbangkan prioritas port 2.
- Tambahkan komponen ketiga (fungsi indikator I) sebagai penalti berat jika SLA Port 4 dilanggar, sesuai konsep *Safe RL* 14, 11.