Università degli Studi di Padova

Relazione per:

Ricerca Operativa

Per Simulare una:

Minimizzare il numero di turni per sconfiggere un Boss nei giochi RPG

Realizzata da:
Gabriele Di Pietro, matricola 2010000

GitHub Repository:
`https://github.com/SerpenTaki/ProgettoRO-2025`

1 Introduzione

1.1 Abstract

Si presenta un problema di ottimizzazione strategica nei giochi RPG dove il giocatore deve comporre un team di personaggi per minimizzare il numero di turni impiegati per uccidere un eventuale boss.

1.2 Descrizione del problema

Per un gioco di ruolo il giocatore può scegliere di comporre un team di personaggi. Ogni *party*[1] è formato da 3 personaggi scelti tra Cavalieri e Maghi. Questo significa che un team può essere formato da 3 Cavalieri, 3 Maghi o da un misto di entrambi. La stamina e il mana sono statistiche separate ma condivise tra i personaggi; in generale il mana viene usato dai Maghi, mentre la stamina dai Cavalieri. I 3 personaggi possono decidere di attaccare durante lo stesso turno o di non attaccare per quello specifico turno per recuperare un certo quantitativo di mana e stamina. I Cavalieri, quando attaccano, possono decidere di usare la spada con 2 mani consumando il 75% di stamina in più ma infliggendo il doppio dei danni. Mentre i Maghi, decidendo di non attaccare, potenziano l'attacco dei Cavalieri del 20% in più per i 2 turni successivi. Non è possibile che 2 Maghi si riposino nello stesso turno. Si vuole minimizzare il numero di turni totali per uccidere un grande mostro con un determinato quantitativo di vita. Sapendo che:

- Un Boss ha un certo numero di punti vita;
- Nei turni di riposo il mago ripristina il 20% di mana consumato;
- Nei turni di riposo il cavaliere ripristina il 30% di stamina consumata;
- Il boost del cavaliere ottenuto dal mago che si riposa aumenta del 20% il danno base con una mano che il cavaliere infligge;

2 Modello

2.1 Insiemi

- T: Insieme dei turni possibili, con $t \in T \subseteq \mathbb{Z}^+$ e nell'implementazione $T = \{1, 2, 3, ..., 20\}$
- K: Insieme dei cavalieri, con $k \in K \subseteq \mathbb{Z}^+$ e $k = \{0, 1, 2, 3\}$
- W: Insieme dei maghi, con $w \in W \subseteq \mathbb{Z}^+$ e $w = 3 - k$

[1] Insieme di cavalieri e maghi che compongono il team.

2.2 Parametri

- PV_{Boss}: punti vita del Boss
- DK: danno che un cavaliere può infliggere
- DW: danno che un mago può infliggere
- SKC: consumo di stamina da parte di un cavaliere
- MWC: consumo di mana da parte di un mago
- S_{max}: stamina massima del team
- M_{max}: mana massimo del team
- S_{min}: stamina minima del team $= 0$
- M_{min}: mana minima del team $= 0$

2.3 Variabili decisionali

- S_t: Stamina disponibile all'inizio del turno
- M_t: Mana disponibile all'inizio del turno
- DPT_t: Danno per turno, quindi il danno totale inflitto dal *party* al turno t
- $x_{k,t} = \begin{cases} 1 & \text{se il cavaliere } k \text{ attacca al turno } t \\ 0 & \text{altrimenti} \end{cases}$
- $y_{k,t} = \begin{cases} 1 & \text{se il cavaliere } k \text{ usa 2 mani per attaccare} \\ 0 & \text{altrimenti} \end{cases}$
- $z_{w,t} = \begin{cases} 1 & \text{se il mago } w \text{ attacca nel turno } t \\ 0 & \text{altrimenti} \end{cases}$
- $Boost_t = \begin{cases} 1 & \text{se un mago si è riposato in uno dei 2 turni precedenti} \\ 0 & \text{altrimenti} \end{cases}$
- $u_t = \begin{cases} 1 & \text{se il boss è stato sconfitto nel turno } t \\ 0 & \text{altrimenti} \end{cases}$

2.4 Funzione obiettivo

$$\min \sum_{t \in T} t \cdot u_t \tag{1}$$

L'obiettivo è quello di trovare il turno minimo in cui il boss viene sconfitto. Quindi sommiamo tutti i turni ma solo quello dove viene sconfitto non va a 0.

2.4.1 Vincoli

La stamina e il mana sono valori che si aggiornano in base al consumo (*durante la fase di attacco*) o alla ricarica (*durante la fase di riposo*).

$S_t \geq S_{min} \qquad S_t \leq S_{max} \qquad M_t \geq M_{min} \qquad M_t \leq M_{max} \qquad \forall t$

$y_{k,t} \leq x_{k,t} \qquad \forall k, t$

$\sum_{k \in K} SKC(x_{k,t} + 0.75 y_{k,t}) \leq S_t \qquad \forall t$

$\sum_{w \in W} z_{w,t} \cdot MWC \leq M_t \qquad \forall t$

$S_{t+1} = S_t - \sum_{k \in K} SKC(x_{k,t} + 0.75 y_{k,t}) + 0.3 \cdot r_{k,t} \qquad \forall k, t$

$M_{t+1} = M_t - \sum_{w \in W} z_{w,t} \cdot MWC + 0.2 \cdot \gamma_{w,t} \qquad \forall w, t$

dove definiamo r e γ come 2 variabili ausiliarie usate per linearizzare il vincolo di riposo dei cavalieri e maghi, e sono soggette a:

$r_{k,t} \leq S_{max}(1 - x_{k,t})$ $\qquad$ $\gamma_{w,t} \leq M_{max}(1 - z_{w,t})$

$r_{k,t} \geq (S_{max} - S_t) - M \cdot x_{k,t}$ $\qquad$ $\gamma_{w,t} \geq (M_{max} - M_t) - M \cdot z_{w,t}$

$r_{k,t} \leq S_{max} - S_t$ $\qquad$ $\gamma_{w,t} \leq M_{max} - M_t$

$r_{k,t} \geq 0$ $\qquad$ $\gamma_{w,t} \geq 0$

Dove M è un numero sufficientemente grande per garantire che il vincolo sia soddisfatto, ad esempio $M = S_{max}$.

Vincoli sul danno e sulla condizione di vittoria

$$Boost_t \leq \sum_{w \in W}((1 - z_{w,t-1}) + (1 - z_{w,t-2})) \qquad \forall t$$

$$z_{m,t-1} + z_{m,t-2} \leq 1 \qquad \forall t$$

$$DPT_t = (\sum_{k \in K} DK \cdot (x_{k,t} + y_{k,t} + (0.2Boost_t))) + \sum_{w \in W} z_{w,t} \cdot DW \qquad \forall t$$

$$\sum_{t=1}^{t} DPT_t \geq PV_{Boss} \cdot u_t \qquad \forall t$$

Vincolo di riposo dei maghi, in un turno solo 1 mago

$$\sum_{w \in W}(1 - z_{w,t}) \leq 1 \qquad \forall t$$

Solo un turno può essere quello in cui il boss viene sconfitto

$$\sum_{t \in T} u_t = 1 \qquad \forall t$$

Domini:

$x_{k,t}, y_{k,t}, z_{w,t}, u_t, Boost_t \in \{0, 1\}$ in $\mathbb{Z}^+$

$S_t, M_t, DPT_t \geq 0,$ con $S_t, M_t, DPT_t \in \mathbb{R}^+$

3 Implementazione

Presento di seguito l'implementazione del modello in AMPL e il relativo file di esecuzione.

3.1 File .run

```
reset;
model file.mod;
data Dati/file.dat; # cambiare il nome del "file.dat" per
    selezionare lo scenario
option solver gurobi;
solve;

display TurniMinimi;
display u, DPT, St, Mt;
```

1: File di esecuzione

3.2 File .mod

```
set T ordered;
set K;
set W;

param PVBoss;
param DK;
param DW;
param SKC;
param MWC;
param Smax;
param Mmax;
param Smin;
param Mmin;
param BigM := 100;

var St{T} >= 0;
var Mt{T} >= 0;
var DPT{T} >= 0;

var x{k in K, t in T} binary;
var y{k in K, t in T} binary;
var z{w in W, t in T} binary;
var Boost{T} binary;
var u{T} binary;

var r{k in K, t in T} >= 0;
var gamma{w in W, t in T} >= 0;

minimize TurniMinimi:
    sum {t in T} t * u[t];

subject to StaminaMin {t in T: t != last(T)}:
    St[t] >= Smin;

subject to StaminaMax {t in T: t != last(T)}:
    St[t] <= Smax;

subject to ManaMin {t in T: t != last(T)}:
    Mt[t] >= Mmin;

subject to ManaMax {t in T: t != last(T)}:
    Mt[t] <= Mmax;
```

```
subject to Attacco2mani{k in K, t in T}:
    y[k,t] <= x[k,t];

subject to ConsumoStamina{t in T}:
    sum{k in K} SKC * (x[k,t] + 0.75 * y[k,t]) <= St[t];

subject to ConsumoMana{t in T}:
    sum{w in W} z[w,t] * MWC <= Mt[t];

subject to AggiornaStamina{t in T: ord(t) < card(T)}:
    St[t+1] = St[t] - sum{k in K} SKC * (x[k,t] + 0.75 * y[k,
   t]) + 0.3 * sum{k in K} r[k,t];

subject to AggiornaMana{t in T: ord(t) < card(T)}:
    Mt[t+1] = Mt[t] - sum{w in W} z[w,t] * MWC + 0.2 * sum{w
   in W} gamma[w,t];

subject to RiposoCavaliere1 {k in K, t in T}:
    r[k,t] <= Smax * (1 - x[k,t]);

subject to RiposoCavaliere2 {k in K, t in T}:
    r[k,t] >= (Smax - St[t]) - BigM * x[k,t];

subject to RiposoCavaliere3 {k in K, t in T}:
    r[k,t] <= Smax - St[t];

subject to RiposoMago1 {w in W, t in T}:
    gamma[w,t] <= Mmax * (1 - z[w,t]);

subject to RiposoMago2 {w in W, t in T}:
    gamma[w,t] >= (Mmax - Mt[t]) - BigM * z[w,t];

subject to RiposoMago3 {w in W, t in T}:
    gamma[w,t] <= Mmax - Mt[t];

subject to BoostCondition {t in T: ord(t) > 2}:
    Boost[t] <= sum {w in W} (1 - z[w,t-1] + 1 - z[w,t-2]);

subject to UnoRiposaPerTurno {t in T}:
    sum {w in W} (1 - z[w,t]) <= 1;

subject to DannoPerTurno {t in T}:
    DPT[t] = sum{k in K} DK * (x[k,t] + y[k,t] + 0.2 * Boost[
   t]) + sum{w in W} z[w,t] * DW;
```

```
subject to BossSconfitto {t in T}:
    sum{tt in T: ord(tt) <= ord(t)} DPT[tt] >= PVBoss * u[t];

subject to SoloUnTurno:
    sum{t in T} u[t] = 1;
```

2: Modello in Ampl

4 Primo scenario e introduzione

Negli scenari presentati di seguito, il file `.dat` è stato modificato per cambiare i parametri del problema. Ipotizzando la scelta che un giocatore può fare per comporre il proprio team. Nel primo scenario, il team è composto da 2 cavalieri ed un mago. Il boss presenta 1200 Punti vita, e le statistiche di attacco e consumo sono le seguenti:

Parametro	Valore
DK	30
DW	25
SKC	20
MWC	15

Per tutti gli scenari queste statistiche sono state mantenute costanti. In modo tale che ci si possa concentrare sulla differenza della composizione del *party*

4.1 File .dat

```
set T := 1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20;
set K := k1,k2;
set W := w1;
param PVBoss := 1200;
param DK := 30;
param DW := 25;
param SKC := 20;
param MWC := 15;
param Smax := 100;
param Mmax := 100;
param Smin := 0;
param Mmin := 0;
var St :=
1 100;
var Mt :=
1 100;
```

4.2 Risultati

L'esecuzione del file `.run` mostra che il numero minimo di turni per sconfiggere il boss è 16. In particolare possiamo notare come appena disponibile il modello predilige l'attacco a 2 mani da parte del cavaliere

5 Secondo scenario

5.1 File .dat

5.2 Risultati

6 Terzo scenario e conclusioni

6.1 File .dat

6.2 Risultati

7 Conclusioni