Comparazione tra Physarum Polycephalum Algorithm Based (PPAB) e A* per la risoluzione di problemi di path-finding

Marco Locatelli[1] and Fabio D'Elia[2]

[1]830667, Computer Science, Università degli studi di Milano-Bicocca,
m.locatelli60 at campus.unimib.it
[2]829937, Computer Science, Università degli studi di Milano-Bicocca,
f.delia2 at campus.unimib.it

November 14, 2021

Abstract

Il Physarum Polycephalum è un organismo unicellulare, simile ad una muffa, di colore giallo, espandibile, organizzata attraverso un'intricata rete di tubi interlacciati. Trova interesse in molti ambiti, come la risoluzione di problemi di attraversamento di grafi, in quanto presenta molte caratteristiche, come: la capacità di imparare dalle proprie azioni e dall'ambiente in cui viene inserito, ed inoltre la capacità di trovare il tragitto più corto verso una fonte di cibo. Lo studio della risoluzione di algoritmi di path-finding, si baserà sulla comparazione di un algoritmo come A* e la nostra variante basata su Physarum Polycephalum.

1 Introduzione

Il path-finding è la ricerca e il tracciamento di un percorso minimale tra un punto di partenza ed un punto di fine. Il problema del path-finding trova molteplici applicazioni, che spaziano dal maze-solving, alle applicazioni in ambito di guida autonoma.

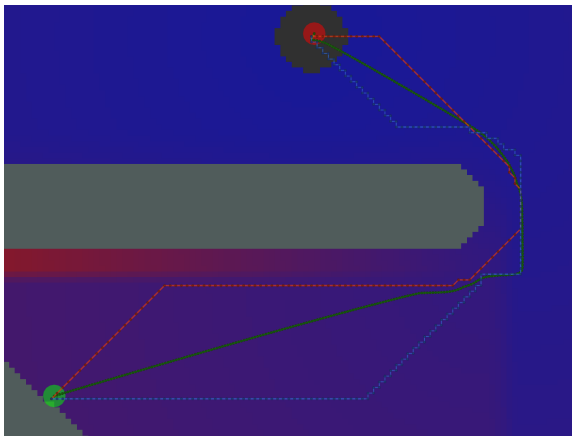


Figure 1: Algoritmi di path-finding applicati per il global planning di un robot in un environment contenente ostacoli.

Il Physarum Polycephalum è un'organismo unicellulare [1] il cui corpo si estende in forma tubolare. Andando a studiare il meccanismo fisiologico di creazione di tubi e della loro distruzione è stato costruito un modello matematico [2].

Basandoci sul modello matematico, è stato implementata una simulazione del Physarum Polycephalum utilizzando un'automa cellulare [3]. Andremo dunque a confrontare due tipologie differenti di algoritmi di path-finding: uno dotato di euristica come A* e la nostra variante basata su automa cellulare del Physarum Polycephalum.

2 Automa Cellulare

Un'automa cellulare è un modello matematico [4] in cui componenti semplici agiscono insieme per simulare comportamenti di sistemi dinamici. Gli automi cellulari sono definiti dalle seguenti caratteristiche [5]:

- La *struttura* è rappresentata con un griglia di celle n-dimensionale. Ad ogni cella è assegnato uno stato che viene aggiornato ad ogni time-step. Lo stato viene aggiornato con una *updating rule* definita sulla base del vicinato.

- Il *vicinato* sono le celle che confinano con una determinata cella, sono fondamentali in quanto partecipano alla definizione dello stato di una cella. I due metodi principali per definire la struttura del vicinato sono: Von Neumann (2) e Moore (3).

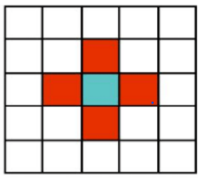


Figure 2: Vicinato di Von Neumann con $r = 1$

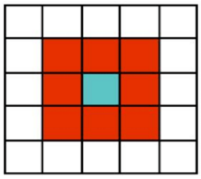


Figure 3: Vicinato di Moore con $r = 1$

- La *condizione dei bordi* è la specifica del comportamento della updating rule e la definizione del vicinato quando si è sui bordi (per esempio i bordi opposti vengono collegati tra loro).
- L' *updating rule* è l'insieme di azioni e regole che definiscono il prossimo stato di una cella sulla base della stessa e del suo vicinato. Questo processo è solitamente identico per ogni tipo di cella, mentre in alcuni casi è una funzione del tempo o della posizione.

3 Modellazione del Physarum Polycephalum

Per la creazione del nostro modello [3] di Physarum Polycephalum, abbiamo basato lo sviluppo sulla definizione di regole che simulassero accuratamente l'evoluzione dell'organismo unicellulare. Il Physarum Polycephalum [1], durante la fase gelatinosa, è composto principalmente da una matrice spugnosa di fibre contrattili di miosina e actina, che generano delle oscillazioni locali. La percezione della variazione di concentrazione di nutriente, causa un'alterazione della membrana del plasmodio, che ammorbidendosi genera un flusso di protoplasma nella direzione della variazione, in risposta a una variazione interna della pressione dovuta alle continue oscillazioni locali. In questo modo, il plasmodio esplora l'area disponibile, incapsulando tutti gli Nutrient Source (NS) e crea una rete di tubi che li collega.

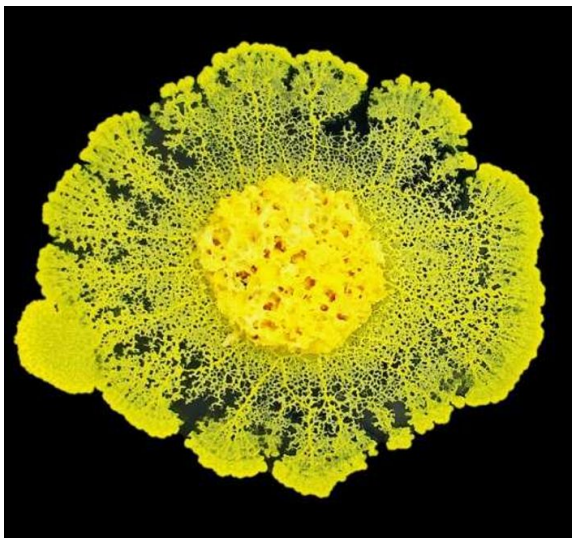


Figure 4: Physarum Polycephalum

Nell'automa cellulare vengono definiti un insieme finito di stati per ogni cella. Nel nostro caso, sono stati definiti 4 stati:

- **A**: Available Cell.
- **U**: Unavailable Cell.
- **NS**: Nutrient Source.
- **SP**: Starting Point.

Per quanto riguarda i primi 2 stati sono fissi durante l'esecuzione della simulazione, mentre gli *NS* diventeranno *SP* una volta incapsulati. È possibile che gli *SP* diventino *NS*, come vedremo successivamente.
Inoltre per ogni cella vengono definiti ulteriori parametri:

- **AA**: Available Area.
- **PM**: Physarum Mass.
- **CHA**: Chemoattractant.
- **TE**: Tube Existence.

AA è un parametro booleano che indica se la cella è disponibile o meno. Per **PM** si intende la massa di organismo presente in una specifica cella in un specifico momento. **CHA** è la quantità di *chemiotattico*, una sostanza che attrae particolari tipi di cellule, presente all'interno di una cella in un momento specifico. Infine **TE** indica la presenza o meno di un tubo in una cella.

3.1 Equazione di diffusione

In questa sezione andremo a spiegare come il PM e il CHA si evolvono step per step all'interno dell'automa cellulare. Ogni cella usa i valori di PM e CHA dei suoi vicini per calcolare il valore corrente di questi parametri, in particolare vengono definite le equazioni per il vicinato di Von Neumann (pmVN) e per quello di Moore (pmMN), in quanto verranno assegnati pesi diversi al risultato di tali equazioni. Ipotizziamo che la cella su cui stiamo eseguendo la simulazione sia la cella di posizione (i, j) allora le equazioni saranno:

$$\begin{aligned} pmVN_{(i,j)} =&((1 + PA_{(i,j)(i-1,j)}) \cdot PM_{(i-1,j)} - AA_{(i-1,j)} \cdot PM_{(i,j)})+ \\ &((1 + PA_{(i,j)(i+1,j)}) \cdot PM_{(i+1,j)} - AA_{(i+1,j)} \cdot PM_{(i,j)})+ \\ &((1 + PA_{(i,j)(i,j-1)}) \cdot PM_{(i,j-1)} - AA_{(i,j-1)} \cdot PM_{(i,j)})+ \\ &((1 + PA_{(i,j)(i,j+1)}) \cdot PM_{(i,j+1)} - AA_{(i,j+1)} \cdot PM_{(i,j)}) \end{aligned} \tag{1}$$

$$\begin{aligned} pmMN_{(i,j)} =&((1 + PA_{(i,j)(i-1,j-1)}) \cdot PM_{(i-1,j-1)} - AA_{(i-1,j-1)} \cdot PM_{(i,j)})+ \\ &((1 + PA_{(i,j)(i+1,j-1)}) \cdot PM_{(i+1,j-1)} - AA_{(i+1,j-1)} \cdot PM_{(i,j)})+ \\ &((1 + PA_{(i,j)(i+1,j+1)}) \cdot PM_{(i+1,j+1)} - AA_{(i+1,j+1)} \cdot PM_{(i,j)})+ \\ &((1 + PA_{(i,j)(i-1,j+1)}) \cdot PM_{(i-1,j+1)} - AA_{(i-1,j+1)} \cdot PM_{(i,j)}) \end{aligned} \tag{2}$$

$$PM_{(i,j)} = PM_{(i,j)} \ + \ PMP1 \ \cdot \ [pmVN_{(i,j)} \ + \ PMP2 \cdot pmMN_{(i,j)}] \tag{3}$$

Come detto precedentemente e come si può notare all'interno dell'equazione (3), i pesi per le due tipologie di vicinato non sono gli stessi.
Si evidenzia che un'area non disponibile (U) non contribuirà al calcolo della massa del Physarum. Inoltre la variabile $PA_{(i,j),(k,l)}$ rappresenta l'attrazione della massa del Physarum nella cella (i, j) verso la direzione della cella adiacente (k, l), in questo modo viene modellata l'attrazione verso il gradiente più alto del CHA. Operativamente assegniamo a $PA_{(i,j),(k,l)}$ una costante, che chiamamo PAP, alla cella adiacente con il più alto valore di CHA e il valore negativo, della stessa costante, alla cella opposta a quella con il CHA con valore maggiore. Per il resto delle celle adiacenti il valore di PA è uguale a 0. Definiamo formalmente il valore di PA per la cella (i, j) verso un vicino $(i - 1, j)$ come mostrato nell'equazione sottostante:

$$PA_{(i,j),(i-1,j)} = \begin{cases} PAP & if \ \ CHA_{(i-1,j)} = MAX(CHA) \\ -PAP & if \ \ CHA_{(i+1,j)} = MAX(CHA) \\ 0 & altrimenti \end{cases} \tag{4}$$

dove il MAX(CHA) viene calcolato tra tutte le celle adiacenti alla cella (i, j).
In aggiunta, come accennato in precedenza, anche il contributo per la diffusione del CHA si basa sul valore del vicinato di Von Neumann e di Moore, come illustrato

nelle equazioni sottostanti:

$$\begin{aligned} CHAVN_{(i,j)} =&(CHA_{(i-1,j)} - AA_{(i-1,j)} \cdot CHA_{(i,j)})+ \\ &(CHA_{(i+1,j)} - AA_{(i+1,j)} \cdot CHA_{(i,j)})+ \\ &(CHA_{(i,j-1)} - AA_{(i,j-1)} \cdot CHA_{(i,j)})+ \\ &(CHA_{(i,j+1)} - AA_{(i,j+1)} \cdot CHA_{(i,j)}) \end{aligned} \tag{5}$$

$$\begin{aligned} CHAMN_{(i,j)} =&(CHA_{(i-1,j-1)} - AA_{(i-1,j-1)} \cdot CHA_{(i,j)})+ \\ &(CHA_{(i+1,j-1)} - AA_{(i+1,j-1)} \cdot CHA_{(i,j)})+ \\ &(CHA_{(i+1,j-1)} - AA_{(i+1,j-1)} \cdot CHA_{(i,j)})+ \\ &(CHA_{(i-1,j+1)} - AA_{(i-1,j+1)} \cdot CHA_{(i,j)}) \end{aligned} \tag{6}$$

$$CHA_{(i,j)} = CON \cdot \{CHA_{(i,j)} + CAP1 \cdot (CHAVN_{(i,j)} + CAP2 \cdot CHAMN_{(i,j)}\} \tag{7}$$

Come succede per la massa del Physarum, se una cella adiacente è una cella non disponibile (U) allora tale cella non contribuirà alla diffusione del CHA. Da notare che la moltiplicazione con il parametro CON simula il consumo del chemotattico da parte del plasmodio.
Partita la simulazione, e quindi la diffusione del PM e del CHA, una volta che il Physarum raggiunge una fonte di cibo (NS) con quantità di massa che superi una predeterminata soglia (ThPM), allora l'SP più vicino e l'NS in questione verranno collegati con un tubo e la cella contenente la fonte di cibo verrà inizializzata come se fosse un nuovo punto di partenza (SP), dal quale partirà una nuova diffusione della massa del Physarum. In questo modo due fonti di cibo potrebbero essere collegate tra di loro.

3.2 Inizializzazione dei parametri

L'inizializzazione dei vari parametri, cella per cella, al tempo 0, varia dallo stato della cella. Appena sotto riportiamo una tabella di come vengono inizializzate le celle dell'automa cellulare.

	A	**U**	**NS**	**SP**
AA	1	0	1	1
TE	0	0	0	0
PM	0	0	0	100
CHA	0	0	100	0

Table 1: Inizializzazione delle celle per ogni stato

Durante l'equazione di diffusione, avviene una fase di re-inizializzazione di alcuni parametri. Vediamo ora le equazioni formali da seguire in caso di inizializzazione o

di re-inizializzazione.

$$PM_{(i,j)} = \begin{cases} 0 & if\ C_{(i,j)} = U \\ 100 & if\ C_{(i,j)} = SP \\ 100 & if\ C_{(i,j)} = NS\ and\ PM_{(i,j)} \geq ThPM \end{cases} \tag{8}$$

Dove per $ThPM$ si intende la soglia da superare per considerare la fonte di cibo assorbita dal Physarum.

$$CHA_{(i,j)} = \begin{cases} 0 & if\ C_{(i,j)} = U \\ 0 & if\ C_{(i,j)} = SP \\ 0 & if\ C_{(i,j)} = NS\ and\ PM_{(i,j)} \geq ThPM \\ 100 & if\ C_{(i,j)} = NS\ and\ PM_{(i,j)} < ThPM \end{cases} \tag{9}$$

3.3 Flusso del modello

In questa sezione vedremo come lavora l'algoritmo passo per passo.

1. Viene inizializzato il modello con almeno un SP e un NS.
2. Si applica la diffusion equation per 50 time-steps.
3. Si controlla che ci sia qualche NS coperto dalla massa del physarium con una determinata quantità di massa. Nel caso non ci fosse nessun NS coperto si riesegue il punto 2.
4. Tutti gli NS coperti con $PM_{(i,j)} \geq ThPM$ sono incapsulati dal plasomodio e quindi si connette l'SP alla fonte di cibo. Per connettere SP e NS, si parte dall'NS incapsulato e il vicino con il PM più alto farà parte del tubo (TE = True). La cella appena selezionata sceglierà, dal suo vicinato, la cella con il PM più alto e quest'ultima cella farà anch'essa parte del tubo. Si va avanti con questo procedimento finché non si raggiungerà una cella con stato SP.
5. Gli NS appena menzionati vengono cambiati in SP e nel caso non fossero passati i 5000 time-steps si ritorna al punto 2. Se non sono presenti altri NS all'interno dell'automa cellulare s'interrompe la simulazione.
6. Nel caso vengano raggiunti i 5000 time-steps le celle SP e NS vengono redefinite tutte in NS, eccezion fatta per la penultima cella NS cambiata in SP prima dei 5000 time-steps. Dopo questa riconfigurazione si esegue per altri 5000 time-steps.

Il punto 6 viene messo perchè, da alcuni esperimenti in laboratorio, dopo una certa quantità di tempo non definita ma sufficientemente grande, il Physarum sembra cambiare la conformazione della sua "rete".

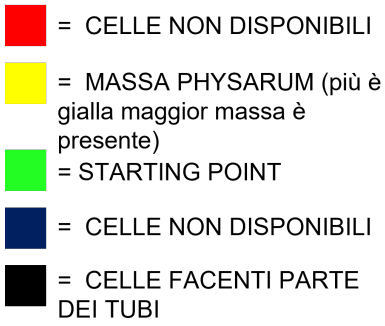


Figure 5: Legenda della rappresentazione grafica.

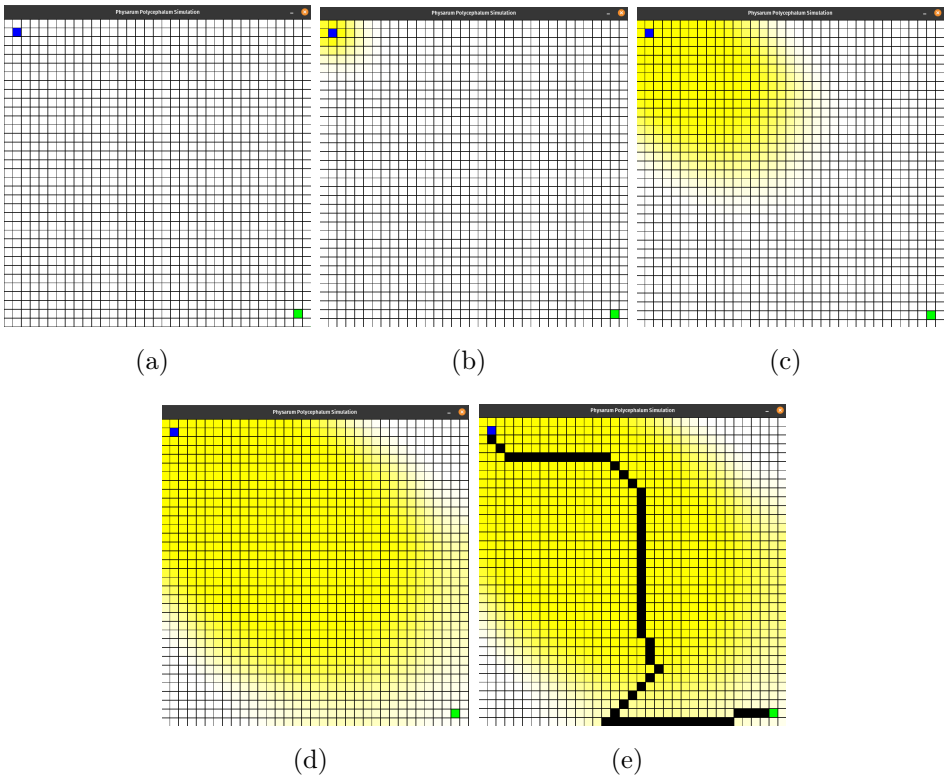


Figure 6: (a) Fase di inizializzazione (b, c) Fase di diffusione (c) Fase di copertura degli NS (e) Fase di costruzione dei tubi

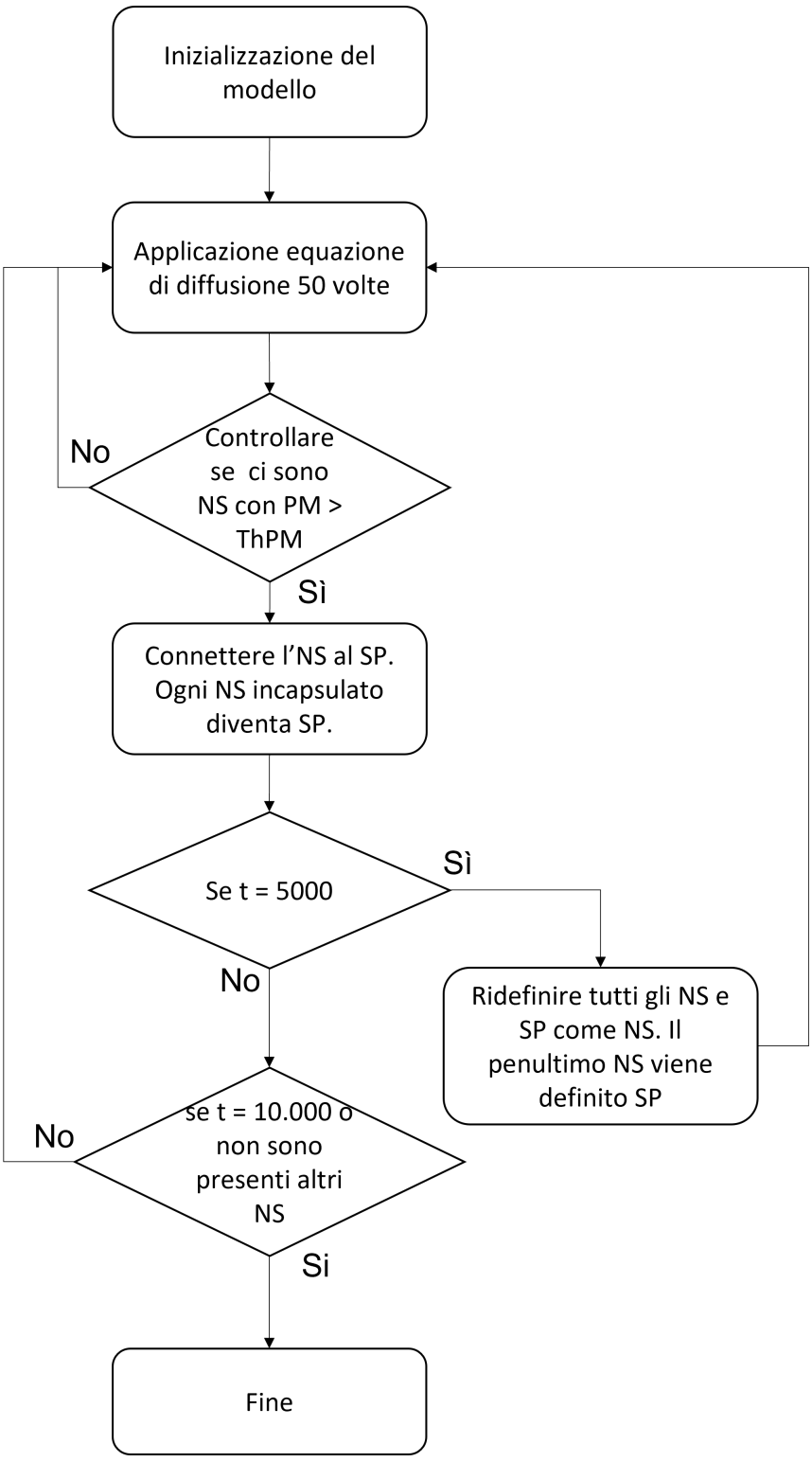


Figure 7: Diagramma di sequenza del flusso dell'algoritmo.

4 Cambiamenti al modello

Il modello appena proposto funziona e i risultati sono più che soddisfacenti; il limite che abbiamo trovato e che abbiamo cercato di migliorare, provando ad ottenere gli stessi risultati, è l'efficienza computazionale. L'idea che sta dietro alla nostra pro-

posta è quella di far seguire al Physarum il gradiente del chemiotattico, spostando la massa del Physarum nelle zone in cui si ha maggior concentrazione a discapito di quelle con meno chemiotattico. Questo meccanismo si basa sul comportamento naturale del Physarum Polycephalum, innestato in un ambiente nuovo in cui è presente una fonte di cibo.

4.1 Flusso del modello

Dato che il nuovo modello si basa fortemente sulla presenza del chemiotattico, abbiamo deciso che per i primi 100 time-steps ci fosse solo la diffusione di CHA nell'automa cellulare e solo successivamente far iniziare la diffusione della massa del Physarum. La diffusione del CHA segue le equazioni (5), (6), (7) viste nel capitolo precedente. La scelta dei 100 time-steps è motivata dal fatto che, con un'automa cellulare 35x35, dimensioni su cui sono stati eseguiti i test, il modello raggiungesse uno stato di buona diffusione del CHA, anche se non ancora completa.
Passati i 100 time-steps inizia la diffusione attraverso *l'equazione di diffusione*. L'equazione di diffusione nella prima parte è rimasta invariata, ovvero si cerca nel vicinato la cella con il CHA più alto, a tale cella viene assegnata una costante (PAP) e alla cella in direzione opposta viene assegnato il valore negato della costante (-PAP), come in (4), a questo punto viene calcolato il valore del PM in accordo con l'equazioni (1), (2), (3). Effettuato questo passaggio la massa del Physarum viene modificata ulteriormente, viene quindi individuato il vicino della cella (i, j) con il chemiotattico più basso, ipotiziamo $(i-1, j-1)$, e una parte di massa di $(i-1, j-1)$ viene aggiunta alla cella (i, j). Da notare che questo procedimento si può attuare se il CHA con il valore più basso tra i vicini è minore rispetto al CHA della cella in analisi. Definiamo formalmente i passaggi.

$$(k, l) = (MIN(CHA))\ \forall k, l : i - 1 \leq k \leq i + 1 \wedge j - 1 \leq l \leq j + 1)$$

$$PM_{(i,j)} = \begin{cases} PM_{(i,j)} + \beta PM_{(k,l)} & if\ CHA_{(i,j)} < CHA_{(l,k)} \\ & \quad \wedge CHA_{(k,l)} \neq 0 \\ PM_{(i,j)} & else \end{cases} \tag{10}$$

$$PM_{(k,l)} = \begin{cases} PM_{(k,l)} - \beta PM_{(i,j)} & if\ CHA_{(i,j)} < CHA_{(l,k)} \\ & \quad \wedge CHA_{(k,l)} \neq 0 \\ PM_{(k,l)} & else \end{cases} \tag{11}$$

Successivamente alla diffusione della massa, come nel modello precedente, continua la diffusione del CHA. Per la diffusione del CHA si seguono sempre le equazioni (5), (6), (7).

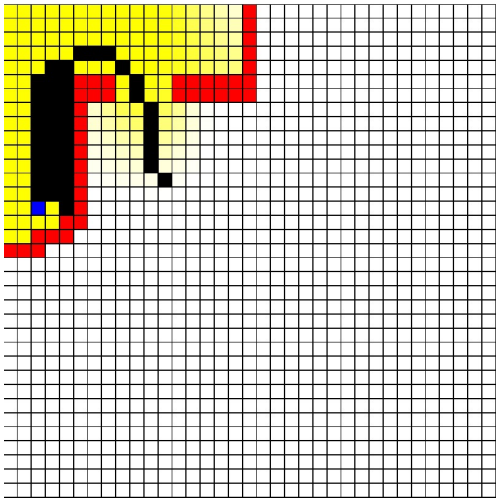


Figure 8: Risultato del nuovo modello.

4.2 Assottigliamento dei tubi

Come possiamo notare nella figura 8 il nuovo modello ha trovato il percorso però con uno spessore dei tubi non ottimale. Ispirati dal comportamento del Physarum [6], che una volta trovato il percorso ottimale mantiene attiva solo la massa che forma i tubi, abbiamo sviluppato l'algoritmo per cercare di ridurre lo spessore dei tubi e trovare il miglior percorso.
Si parte da una situazione come mostrata in figura 8 con tubi temporanei e s'inizia dalla cella contenente il nutrimento. L'algoritmo controlla se lo spessore del tubo è 1, in tal caso va avanti fino alla cella che contiene uno spessore del tubo maggiore di 1 oppure fino alla cella SP. Nel caso si verifichi lo spessore maggiore di 1 l'algoritmo è fatto in modo che venga preferita la direzione che è stata seguita fino a quel punto. Per esempio, ipotizziamo di essere nella cella (i, j) e che la penultima cella aggiunta ai TE sia $(i-1, j)$ allora la cella che verrà preferita per essere aggiunta a far parte dei tubi è la cella $(i+1, j)$ però solo se essa fa parte dei tubi temporanei. Nell'ipotesi che non si verificasse anche questa opzione allora si tende a preferire le celle contenenti tubi che vanno nella direzione del SP. Per esempio, se partiamo da un NS in basso a sinistra, (i, j), e la cella SP è in alto a sinistra, $(i - k, j)$, l'algoritmo preferirà le celle che vanno verso l'alto e, in particolar modo, quelle che rimangono sulla stessa colonna (j). Altrimenti viene scelto casualmente un vicino facente parte delle celle con tubi temporanei.

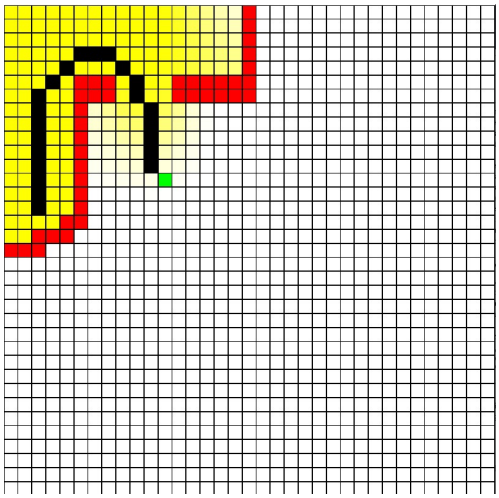


Figure 9: Risultato dell'algoritmo di assottigliamento con partenza dalla figura 8

5 Confronto tra Physarum Polycephalum e A*

A*[7] è un algoritmo di ricerca su grafi e path-search. Si presenta come una variante dell'algoritmo di Dijkstra. A differenza di Dijkstra, A* assegna un peso ad ogni nodo, che è uguale al peso dell'arco sommato alla distanza approssimata tra il nodo considerato e il nodo di arrivo. Con l'utilizzo di questa euristica, si ha un netto miglioramento rispetto all'algoritmo di Dijkstra.

Abbiamo deciso di confrontare A* e la nostra implementazione di Physarum Polycephalum utilizzando diverse ambientazioni, in modo da evidenziare pregi e difetti di entrambi gli algoritmi. Per eseguire il confronto, valuteremo la lunghezza del path trovato ed il tempo utilizzato per computarlo.

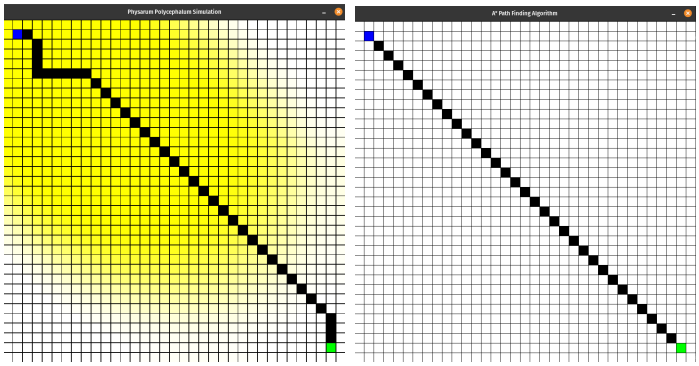
Per quanto riguarda il Physarum Polycephalum, abbiamo utilizzato una configurazione che fosse verosimile, ma che potesse anche competere con A*, dunque:

- PMP1 = 0.08
- PMP2 = 0.01
- CAP1 = 0.05
- CAP2 = 0.02
- CON = 0.95
- PAP = 0.2
- ThPM = 0.2

5.1 Prima ambientazione

Come primo test, abbiamo deciso di posizionare lo starting point, che nel caso del Physarum Polycephalum sarebbe il punto di innesto del fungo, e l'ending point, ovvero il Nutrient Source, ai due vertici opposti dell'ambientazione senza nessun

ostacolo nel mezzo. Così facendo si evidenzia la presenza dell'euristica in A* contrapposta al naturale movimento di espansione del Physarum Polycephalum.



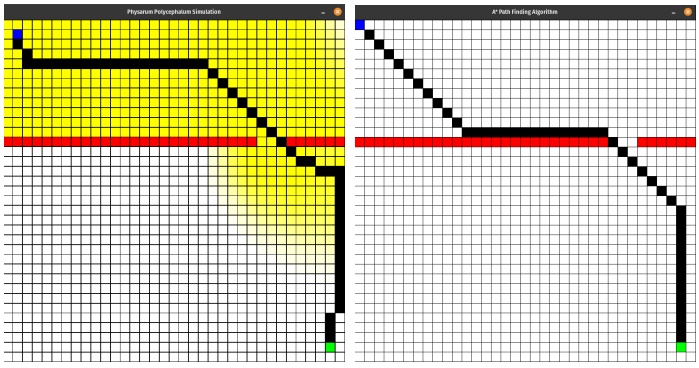
(a) (b)

Figure 10: (a) Path usando Physarum Polycephalum (b) Path usando A*

Come possiamo notare, l'euristica di A* permette di generare il path ottimale, esplorando il minor numero di celle. Il path ottenuto è una linea retta che congiunge i due capi, lunga 32 unità. Contrariamente, il Physarum, necessita di una esplorazione dell'ambiente iniziale, espandendosi in direzione del Nutrient Source (NS). Il path risultante, non è dunque quello ottimale, ma si avvicina di molto. La lunghezza complessiva è di 37 unità. Confrontando i tempi, A* computa il path in circa 2.5 secondi, mentre il Physarum Polycephalum, rallentato dalla fase di esplorazione iniziale, impiega circa 4.4 secondi.

5.2 Seconda ambientazione

Come secondo test, abbiamo introdotto degli ostacoli all'interno dell'ambientazione. In questo caso, abbiamo deciso di bloccare la strada in modo orizzontale, aprendo un piccolo varco posizionato in modo non centrale. L'intento era quello di evidenziare la capacità del Physarum di adattarsi all'ambiente seguendo il gradiente.



(a) (b)

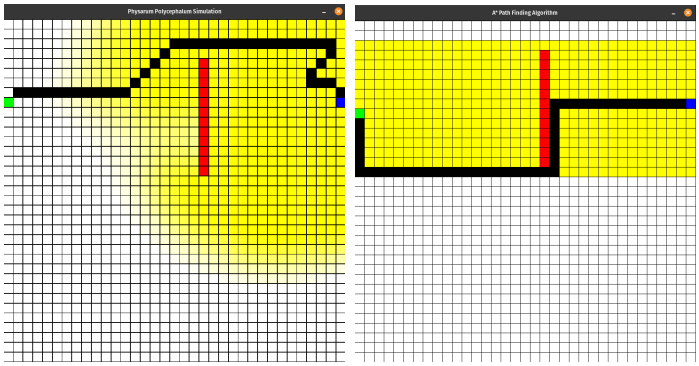
Figure 11: (a) Path usando Physarum Polycephalum (b) Path usando A*

Come si evince, le differenze sui path computati sono minime. Per quanto riguarda il Physarum Polycephalum, la lunghezza del path è di 52 unità, mentre per

A* la lunghezza del path ottenuto è di 46 unità. Per quanto riguarda il tempo di computazione, A* ha impiegato meno della metà del tempo rispetto l'implementazione con Physarum Polycephalum, ovvero circa 4.7 secondi per A* e circa 10.4 secondi per il Physarum Polycephalum.

5.3 Terza ambientazione

Come terzo test, abbiamo deciso di confrontare le due implementazioni in una situazione in cui l'euristica di A* non favorisce una veloce ricerca del path.



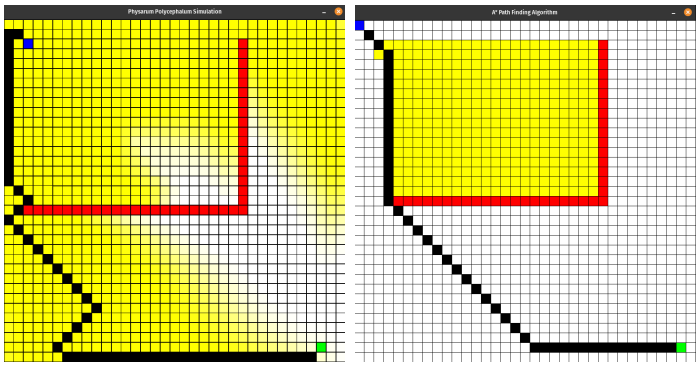
(a) (b)

Figure 12: (a) Path usando Physarum Polycephalum (b) Path usando A*

Il Physarum Polycephalum, durante la fase di esplorazione, ha inizialmente evitato l'ostacolo per poi ricongiungersi, subito dopo, sulla traccia lasciata dal chemiotattico. Contrariamente, A* si trova bloccato di fronte all'ostacolo, dovendo esplorare l'intera ambientazione circostante. Il path risultante da A* è di lunghezza 46 unità computato in circa 11 secondi, mentre il path risultante dall'esplorazione con Physarum è di lunghezza 40 unità computato in circa 7.9 secondi.

5.4 Quarta ambientazione

Come nella terza ambientazione, anche in questo caso abbiamo deciso di introdurre degli ostacoli in modo da indurre A* ad una fase esplorativa prolungata, evidenziando la capacità del Physarum Polycephalum di sfuggire a zone in cui rimarrebbe bloccato.



(a) (b)

Figure 13: (a) Path usando Physarum Polycephalum (b) Path usando A*

Anche in questo caso, è evidente come il Physarum riesca a sfuggire agli ostacoli, computando il path in circa 8 secondi, rispetto ai circa 14.9 secondi di A*, che rimane bloccato in esplorazione nell'area sottesa dagli ostacoli, prima di trovare il path verso l'ending-point. Per quanto riguarda la lunghezza del path, il Physarum Polycephalum traccia un path lungo 61 unità, mentre A* in 47 unità.

5.5 Tabella dei risultati

	Lunghezza	Tempo	Ambientazione
A*	32 u	2.5 s	Ambientazione 1
Physarum Polycephalum	37 u	4.4 s	
A*	46 u	4.7 s	Ambientazione 2
Physarum Polycephalum	52 u	10.3 s	
A*	46 u	11.0 s	Ambientazione 3
Physarum Polycephalum	40 u	7.9 s	
A*	47 u	14.9 s	Ambientazione 4
Physarum Polycephalum	61 u	8.0 s	

6 Conclusione

Abbiamo esplorato le abilità del Physarum Polycephalum nella ricerca di un path tra il punto di innesto e la fonte di cibo.

Dal confronto, abbiamo notato che non sempre A* riesce a computare il path più corto, nel tempo minore, essendo molto legato all'utilizzo di euristiche. Diversamente il Physarum Polycephalum, pur essendo un organismo biologico molto semplice, si comporta in modo egregio in tutte le situazioni.

Inoltre, A* è in grado di trovare path che abbiano un solo starting point ed un unico ending point, mentre il Physarum Polycephalum è abile nel costruire sia path semplici, che reti complesse[8].

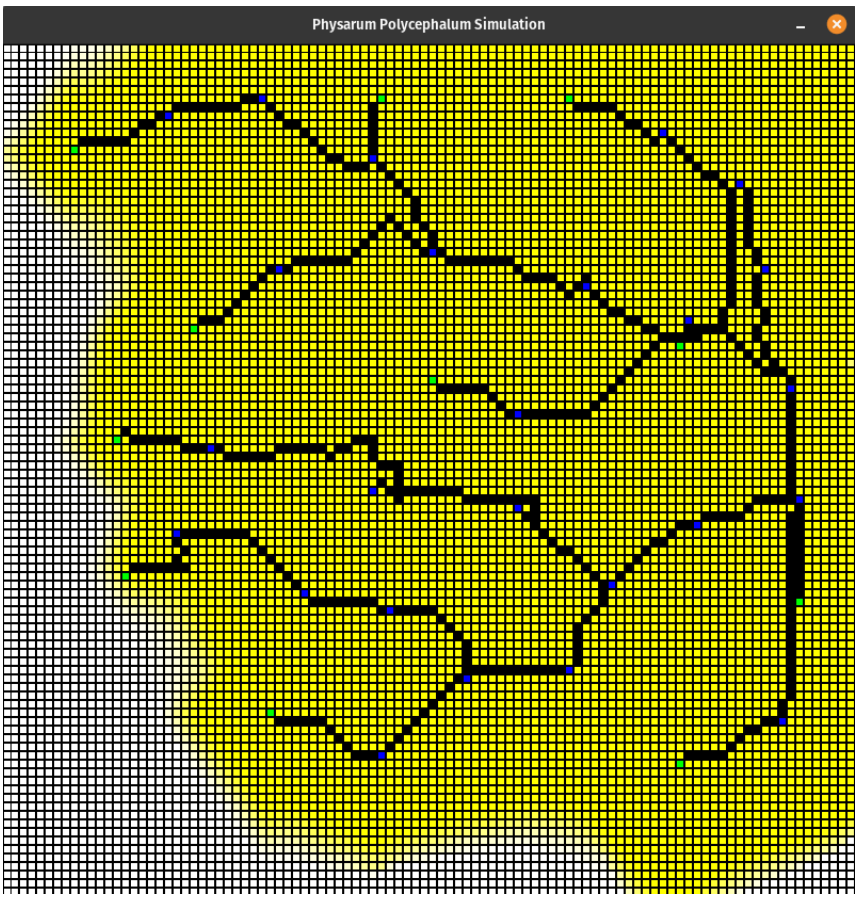


Figure 14: Complessa rete creata dal Physarum Polycephalum, composta da un SP e molteplici NS

6.1 Sviluppi futuri

Dall'esperienza accumulata implementando la nostra variante di Physarum Polycephalum, abbiamo conosciuto le potenzialità della simulazione di modelli biologici applicati in contesti differenti. Si è pensato ad uno sviluppo futuro che emulasse il comportamento dell'organismo biologico, superando la semplice simulazione con l'integrazione di euristiche ed elementi artificiali, con l'intento di migliorarne i risultati ottenuti.

Abbiamo delineato due possibili sviluppi futuri, che sono:

- *Segmentazione di path semplici*, ovvero segmentare il path in diversi parti, aggiungendo NS in modo da caratterizzare ogni parte del path. In questo modo ci aspetteremmo una diffusione del chemiotattico in modo più rapido.

- *Modello di interazione multi-agente*, ovvero lo sviluppo di un modello multi-agente, in grado di comunicare, la presenza di altri agenti o di NS, e collaborare, prestandosi porzioni di massa, al fine di raggiungere gli NS assegnati ad ogni agente.

References

[1] Richard Mayne. Biology of the physarum polycephalum plasmodium: preliminaries for unconventional computing. In *Advances in Physarum machines*, pages 3–22. Springer, 2016.

[2] Atsushi Tero, Ryo Kobayashi, and Toshiyuki Nakagaki. A mathematical model for adaptive transport network in path finding by true slime mold. *Journal of theoretical biology*, 244(4):553–564, 2007.

[3] Michail-Antisthenis I Tsompanas, Georgios Ch Sirakoulis, and Andrew Adamatzky. Cellular automata models simulating slime mould computing. In *Advances in Physarum Machines*, pages 563–594. Springer, 2016.

[4] Stephen Wolfram. Cellular automata as models of complexity. *Nature*, 311(5985):419–424, 1984.

[5] Norman H Packard and Stephen Wolfram. Two-dimensional cellular automata. *Journal of Statistical physics*, 38(5):901–946, 1985.

[6] Xiaoge Zhang, Yajuan Zhang, Zili Zhang, Sankaran Mahadevan, Andrew Adamatzky, and Yong Deng. Rapid physarum algorithm for shortest path problem. *Applied Soft Computing*, 23:19–26, 2014.

[7] Peter E. Hart, Nils J. Nilsson, and Bertram Raphael. A formal basis for the heuristic determination of minimum cost paths. *IEEE Transactions on Systems Science and Cybernetics*, 4(2):100–107, 1968.

[8] Atsushi Tero, Seiji Takagi, Tetsu Saigusa, Kentaro Ito, Daniel Bebber, Mark Fricker, Kenji Yumiki, Ryo Kobayashi, and Toshiyuki Nakagaki. Rules for biologically inspired adaptive network design. *Science (New York, N.Y.)*, 327:439–42, 01 2010.