Navigation With Decision Subspace Constraint

Qin Haocheng 11911317

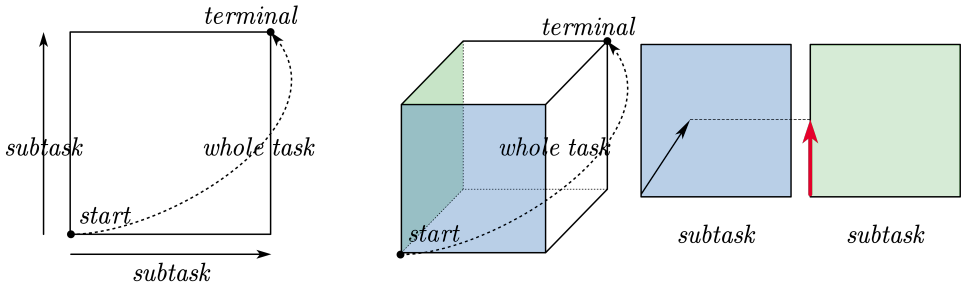
June 15, 2022

Abstract

In this paper, we look into a trivial case that navigation in 2D lattice with only vertical or horizontal links, and get to know the optimized α^* is the same as 1D case, that is 1. We then extend it to a more general case, which the whole space to be $\mathbb{R}^m$ while limit the links to generate with a $\mathbb{R}^d$ subspace constraint for each node, get quite acceptable results for simulation on a extended "circle". Finally, We extent this theorem to a more general case, which the space and constraint can not be $\mathbb{R}^k$, or even not Euclid.

1 Synthesis

Given the problem to find the optimized α for 2D lattice navigation with the constraint that we can only travel vertically or horizontally, Intuitively, it's two subtask in 1D chain, because for step from (0,0) to (n,n), we can just reorder it to a group of vertical steps and a group of horizontal steps, then we are actually doing this: navigate from 0 to n in x axis, then from 0 to n in y axis, like figure 1(a) does. So it's actually an 1D chain problem.



(a) *Trivial case* (a) *Nontrivial Case*

Figure 1: Institution

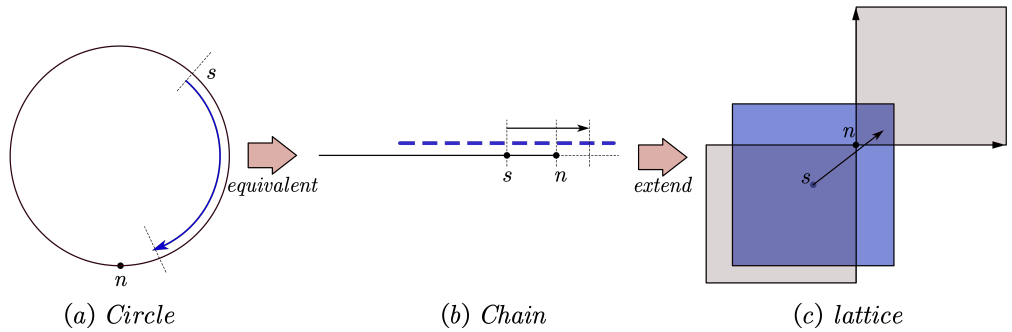
Also, intuitively, if we look into the problem that navigate from $(0,\ldots,0)_m$ to $(n,\ldots,n)_m$ in $\mathbb{R}^m$, while each step we can only travel in (i.e. generate a link in) $\mathbb{R}^d$ subspaces, if the whole space is a direct sum of these subspace, we can similarly claim that it's a d dimensional problem.

However, it's not as trivial as the situation when $d = 1$, because these subspace can have a nonempty interaction. So a link in one subspace may also mean a projected "link" in another subspace, as figure 1(b) shows, i.e. we can no longer view the whole task as a series of independent subtasks.

But actually the assumption is still true for $d > 1$, we will show this both theoretically and practically. It can even not be an Euclid space.

And here, we need to clarify the assumptions for both proof and simulation. We know an easy idea is to keep the scale of distance distribution constant, that is $\sum k^{-\alpha} = const$. For 1D case, we achieve this by simulate in a circle. Thus our idea is to extend "circle" to a higher space.

Actually, a circle is just equivalent to that, in a chain, each node can link to node within a constant distance bound cN, and can be out of the actual bound, as figure 2(a),(b) show.



(a) *Circle* (b) *Chain* (c) *lattice*

Figure 2: Institution

Similarly, for nD cases, we allow each step to jump within a constant bounded hypercube (following the constraint of course) and can be out of the actual bound.

While for convenience for theoretical analysis, we just allow the target to be in the dual hypercube of $[0, N]^n$ about the terminal and the actual bound cube. But note that the scale of distribution is still computed on the whole cube.

2 Theoretical Analysis

Firstly, we want to prove our institution that, the whole task in somehow equivalent to some same subtasks, and these subtasks are easy to be optimized. The difficulty here is to solve the co-move among the interacted subspaces. To solve this, a natural idea is to consider the symmetric to help compute expectation.

Theorem 2.1. *For navigation in $\mathbb{R}^m$ with decision space to be $\mathbb{R}^d \subset \mathbb{R}^m$, the procedure to find an optimized α^* is equivalent to find witch in navigation in $\mathbb{R}^d$ with a full decision space.*

Proof. We now define J_s to be the "jump" vector from position s in the former procedure, $\mathbb{R}^m \simeq U_1 \oplus U2 \oplus \cdots \oplus U_{\binom{m}{d}}$ for $\forall U_i \simeq \mathbb{R}_d$, then the projection $J_s|_{U_i} := J_s^i$. Easy to see that $J_s = \sum J_s^i$.
And because the decision space is $\mathbb{R}^d$, so each step we independently take a jump in arbitrary U_i, so $E\alpha(J_s) = \sum E\alpha(J_s^i)$.
On the other hand, we define I_s to be the "jump" in the later procedure while we express all the vectors in $\mathbb{R}^m$. Obviously, $E\alpha(J_s^i) = \frac{1}{\binom{m}{d}} E_\alpha(I_s^i)$. Here we use the index i to verify different subspace projection of s.
We now define the expected steps to be T_α^J and T_α^I. To compute T_α^I, let $\mathbb{R}^m \simeq E_1 \oplus E_2 \oplus \cdots \oplus E_{\binom{m}{d}}$ for $\forall E_i \simeq \mathbb{R}$. If we want s to achieve the terminal n, we need the projection of s to achieve projection of n in $\mathbb{R}$ for $\forall E_i$, let q_i to be the possibility that it satisfies in E_i for the latest. Then:
$\because \frac{ds}{dt} = E_\alpha(I_s)$
$\Rightarrow T_\alpha^I = \sum q_x \int 1/E_\alpha(I_s)|_{E_x} ds$
For U_j, we have $\{U_t\}_l$, $\forall U \in \{U_t\}_l$, $U \cap U_j \simeq \mathbb{R}^l$
Obviously, $\{U_t\}_{l_1} \cap \{U_t\}_{l_2} = \varnothing$, for $l_1 \neq l_2$
Let L_s^i to be the "jump" of the resulted projection from position s in U_i, then,
$L_s^i = J_s^i + \sum\limits_{2d-m \le l < d} \sum_{\{U_t\}_l} \frac{l}{d} J_s^t|_{U_i}$
For symmetric, we claim $L_s^i = J_s^i + \sum\limits_{2d-m \le l < d} \binom{d}{l}\binom{m-d}{d-l} \frac{l}{d} J_s^i$
Similarly, we know for s to achieve n in procedure J, it needs to achieve each projection in U_i, and the possibility for it to satisfies the latest in U_x is $\frac{1}{\binom{m}{d}}$

$\therefore T_\alpha^J = \sum \frac{1}{\binom{m}{d}} T_\alpha^{J^i} = \sum q_x \int \frac{\binom{m}{d}/(1+\sum_{2d-m\le l<d}\binom{d}{l}\binom{m-d}{d-l}\frac{l}{d}J_s^i)}{E_\alpha(I_s)|_{E_x}} ds$
$= [\binom{m}{d}/(1+\sum_{2d-m\le l<d}\binom{d}{l}\binom{m-d}{d-l}\frac{l}{d}J_s^i)]T_\alpha^I \propto T_\alpha^I$
$\therefore Argmin_\alpha\{T_\alpha^J\} = Argmin_\alpha\{T_\alpha^I\} = \alpha^*$ □

We now only need to consider the common case in the subspace, which was already solved by many work, and we apply an effcient one [1]. To compute the actual α^*, we shall make a special definition for distance. We know we only allow nodes to be in the cube $[0,n]^d$, and its dual to $\vec{n} = (n, n, \ldots, n)$, we then define the distance to the terminal $\vec{n}$ to be $k(j) = sign(Cos(\vec{n}-\vec{j}, \vec{n}))|\vec{j}-\vec{n}|$.

Theorem 2.2. *For navigation in $\mathbb{R}^d$ with a full decision space, $\alpha^* = d$ when d small.*

Proof. $p(k) \sim |k|^{-\alpha}|k|^{d-1} = |k|^{-\alpha+d-1}$, let $A = \sum_{-n^m}^{n^m} |k|^{-\alpha+d-1}$.
Note that this is an approximation, for the true distribution is $p(k,l) \sim |k|^{-\alpha}|l + k \ Mod \ n|^{d-1}$, which varies for different position. So for convinient we take such approximation, but for $d = 1$, it's **exact**, for d small, it's an acceptable approximation.
Then we can write the recurrence for expected steps T_l at one position l away form $\vec{n}$. that is,
$T_l = A\sum_1^{2l-1} k^{-\alpha+d-1}(1 + T_{|l-k|}) + (1 - A\sum_1^{2l-1} k^{-\alpha+d-1})(1 + T_{l-1})$
let $D_l = T_l - T_{l-1}$
$\Rightarrow \frac{d^2T}{dl^2} := D(l) \approx D_{l+1} - D_l \propto \sum_1^l [(l+1-k)^{-\alpha+d-1} - (l+k)^{-\alpha+d-1}]D_k - \sum_1^{2l-1} k^{-\alpha+d-1}D_l$
Approximately, we then have
$T_\alpha \sim n^x$, $x = \begin{cases} \frac{d-\alpha}{d+1-\alpha} & 0 \le \alpha < d \\ \alpha - d & d < \alpha < d+1 \\ 1 & \alpha > d+1 \end{cases}$
For T_α continous for α, we then get $\alpha^* = d$ □

Thus we get that, for navigation in $\mathbb{R}^m$ with decision space to be $\mathbb{R}^d \in \mathbb{R}^m$, the optimized $\alpha^* = d$. And in our case $m = 2$, $d = 1$, so $\alpha^* = 1$.

3 Simulation

For simulation, we follow the setting in synthesis. To verify, we test cases for $m = 2, d = 1$, which is our primary task, and $m = 3, d = 2$, which is a more general case, additionally. i.e.

Algorithm 1 Navigation with constraint

Input: mD Lattice size N^m,clustering exponent α, starting point x_0, terminal x_f.
Output: *step*/mN
  step $\leftarrow 0$, current state $x \leftarrow x_0$
  while $x \neq x_f$ **do**
    Select one distance d from distribution $Pr_N^\alpha(d|x) = d^{\alpha-m+1}/\sum_{i=1}^N i^{\alpha-m+1}$
    Choose one t for all t satisfies $||x-t||_{L_1} = d$ and $\#\{Nonzero\ positions\ in\ x-t\} \le d$ uniformly.
    if $||x_f - t||_{L_1} < ||x_f - x||_{L_1}$ **then**
      $x \leftarrow t$
    else
      x moves one step towards x_f
    end if
    step$+ = 1$
  end while

The results for both cases are:

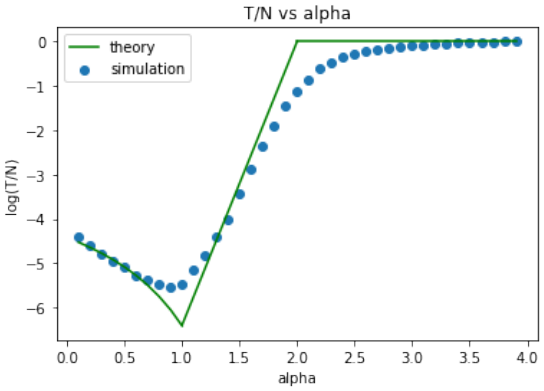


Figure 3: Case m=2, d=1

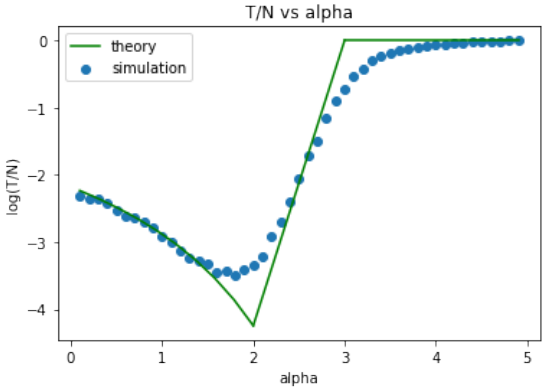


Figure 4: Case m=3, d=2

We found the simulation results follow well with theory in section 3. For $d = 1$, α^* is exactly 1,; for $d = 2$, α^* is slightly smaller than 2, the reason is as the proof on theorem 2.2 says, the larger d is, the worse the approximation to be. But for $d = 1$ we don't suffer this problem, so we can just accept $\alpha^* = 1$, both theoretically and practically.

4 Further Looking

Notice that in proof of theorem 2.1, unlike the introduced work [2], we don't need to bother with the actual distance distribution here, which makes us able to extent this theorem to a more general form. And this is meaningful, for we sometimes care about non-Euclidean tasks, like navigation in a sphere with latitude and longitude lines, with link constraint to be along lines.

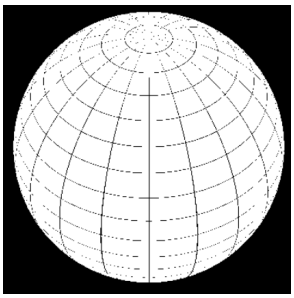


Figure 5: Non-Euclidean Case

For such cases, we can generalize theorem 2.1 as:

Theorem 4.1. *For navigation in target space $\mathcal{U} = U_1 \oplus U2 \oplus \cdots \oplus U_k$ with decision space $\mathcal{V} = \bigcup_1^k U_i|_{\mathcal{U}} \subset \mathcal{U}$, for each step we take a strategy from $\forall U_i$ equally to form a path $\mathcal{L}$ with distance measure $\mathcal{D}$ with parameter α. If:*
1. $U_1 \simeq U_2 \simeq \cdots \simeq U_k$.
2. Existing rotation f, $\forall u \in U_i$ and arbitrary set $\{U_t\}_K := \{U_t \cap U_i \simeq K | \forall U_t\}$, group $\{U_t\}_K = \{f^k(U_i)\}_{k=1}^s$, and always have $\sum f^k(u)|_U \parallel u$ and $||f(u)|_U||_{\mathcal{D}} = ||f^2(u)|_U||_{\mathcal{D}} = \cdots = ||f^k(u)|_U||_{\mathcal{D}}$.
3. Existing at least one finite $\mathcal{L}$ from start to terminal, and the set $\{\mathcal{L}\}$ is independent from $\mathcal{D}$.
Then if we call the expected step for task above to be $T(\mathcal{U}, \mathcal{V}, \mathcal{D}, \alpha)$, there is:
$Argmin_\alpha T(\mathcal{U}, \mathcal{V}, \mathcal{D}, \alpha) = Argmin_\alpha T(\mathcal{V}, \mathcal{V}, \mathcal{D}, \alpha)$.

Then we can simplify the problem. So actually our task is just a special case in Euclid space and a special special case for $\mathcal{U} = \mathbb{R}^2$ and $\mathcal{V} = \mathbb{R}$.

References

[1] S. Carmi, S. Carter, J. Sun, and D. Ben-Avraham, "Asymptotic behavior of the kleinberg model," *Physical review letters*, vol. 102, no. 23, p. 238702, 2009.

[2] L. Barriere, P. Fraigniaud, E. Kranakis, and D. Krizanc, "Efficient routing in networks with long range contacts," in *International Symposium on Distributed Computing*, pp. 270–284, Springer, 2001.