Finding the Longest Isometric Cycle in a Graph

Daniel Lokshtanov *

Abstract

A cycle in a graph G is *isometric* if the distance between two vertices in the cycle is equal to their distance in G. Finding the longest isometric cycle of a graph is then a natural variant of the problem of finding a longest cycle. While most variants of the longest cycle problem are NP-complete, we show that quite surprisingly, one can find a longest isometric cycle in a graph in polynomial time.

1 Introduction

For a given connected graph G, determining whether the graph contains a cycle of a certain kind is one of the most fundamental and basic problems in combinatorics. Probably, the first such problem to be studied was the famous "Seven Bridges of Königsberg". This problem asks whether it is possible to follow a route through the city of Königsberg such that one crosses each of the seven bridges exactly once. In 1736, Euler proved that this is in fact impossible, by characterizing the graphs that admit an "Eulerian walk". An Eulerian walk is a cyclic walk through the graph that passes each edge exactly once, and Euler showed that such a walk exists if and only if every vertex of the graph has even degree [4]. While this problem was the first in the line of many problems about cycles in graphs, it remains one of the quite few such problems that has been shown computationally tractable. A very natural generalization of an Eulerian walk is a Hamiltonian cycle. A Hamiltonian

*Department of Informatics, University of Bergen, N-5020 Bergen, Norway. Email: daniello@ii.uib.no.

cycle is a cycle that passes through each vertex exactly once. While they are similar in formulation, there is a dramatic difference in the computational tractability of these problems. In fact, after the concept of $\mathcal{NP}$-completeness was defined by Cook and Levin, determining whether a given graph G has a Hamiltonian cycle appeared on Karp's list of the 21 first $\mathcal{NP}$-complete problems [2] . Following this, it has been shown that finding the longest cycle and longest induced cycle in a graph are $\mathcal{NP}$-complete problems. [3]

In this paper we consider another variant of the longest cycle problem. Suppose we are to arrange a race in a graph G, and that we want the start and the finish to be in the same place. Furthermore, we are aware that the participants are notorious cheaters, and we want to make sure there are no shortcuts the participants can follow. Finally, we wish the race track to be as long as possible. This amounts to finding the longest *isometric* cycle in G, that is, the longest cycle such that for any two vertices on the cycle, the distance between them in the cycle is equal to their distance in the graph. At the first glance, this variant seems to resemble the longest induced cycle problem, and thus to be intractible unless $\mathcal{P} \neq \mathcal{NP}$. We show that contrary to this intuition, the longest isometric cycle of a graph ca nbe found in polynomial time. The idea behind our algorithm is that isometric cycles are somehow "rigid". Informally, we show that a necessary and sufficient condition for a cycle of length k to be isometric is that the distance in G between every pair of diametrically opposite vertices in the cycle is $\frac{k}{2}$.

2 Definitions and terminology

All input graphs are simple, connected, unweighted and undirected. A *walk* W is a sequence of vertices where each consecutive pair of vertices is connected by an edge. If the first and last vertex of W are the same we say that W is *cyclic*. If all vertices in W are unique we say that the walk is a *path*. If W has at least 3 vertices and all vertices of W are unique, but the first and last vertex are the same, W is a cycle. The *length* of a walk is the number of edges in it. The number of edges in the shortest path between two vertices u and v in a graph G is denoted $d_G(u, v)$ and is called the *distance* between u and v. When the graph is not specified we implicitly mean distance in G and write $d(u, v)$ for short. If u and v actually are the same vertex, we say that $d_G(u, v) = 0$. If u and v lie in different components of G, the distance between them is infinite. For a natural number p, *G to the power of p* is the graph $G^p = (V(G), \{(u,v) : d_G(u,v) \leq p\})$. A subgraph H of a graph G is an *isometric* subgraph, if for every u and v in $V(H)$, $d_H(u,v) = d_G(u,v)$. Notice that an isometric subgraph is an induced subgraph.

3 A useful auxiliary graph

In this section we are going to concentrate upon an auxiliary graph that we can use to test whether a given graph G has an isometric cycle of length exactly k. Assume that $k \geq 3$. Using G we build a new graph G_k. The set of vertices of G_k is the set of ordered pairs $\{(u,v) \in V : d(u,v) = \lfloor k/2 \rfloor\}$ and there is an edge between (u,v) and (w,x) in G_k if $(u,w) \in E(G)$ and $(v,x) \in E(G)$. In order to use G_k we must say something about the relationship between distances in G and distances in G_k.

Lemma 3.1 $d_{G_k}((u,v),(q,w)) \geq \max\{d(u,q), d(v,w)\}$.

Proof. Let $P = \{(u,v),(a_1,b_1),(a_2,b_2)...(q,w)\}$ be a shortest path from (u,v) to (q,w) in G_k. Then both $P_1 = u, a_1, a_2...q$ and $P_2 = v, b_1, b_2...w$ are paths in G. P, P_1 and P_2 all have the same length, completing the proof. ∎

The intuition behind the following results is that a particular cycle of length k is an isometric cycle if and only if the distance in G between every pair of diametrically opposite vertices u and v in the cycle is $\lfloor \frac{k}{2} \rfloor$. The condition is clearly necessary, but it is not obvious that it is sufficient. The reason that the condition is sufficient is that if the cycle is not isometric, one can find a pair of vertices u and v such that their distance in the graph is smaller than the distance in the cycle. If we then look at the pair u and x where x is a vertex in the cycle diametrically opposite to u, such that v lies on the shortest path from u to x in the cycle, we can observe that the distance between u and x in the graph is strictly less than $\lfloor \frac{k}{2} \rfloor$ because we can take the shortcut to v in the graph on the way to x.

The idea now is that the vertices of G_k represent pairs of vertices that might end up as diametrically opposite vertices in an isometric cycle of G, and that two vertices in G_k are adjacent if the pairs thay represnt could be adjacent pairs of diametrically opposite vertices in an isometric cycle of G. Let (u,v) be a vertex in G_k. Notice that such a vertex must exist if there is an isometric cycle of length k in G. The following results formalize the above discussion, and will allow us to use G_k as a tool for finding isometric cycles of length k in G.

Lemma 3.2 *If k is even and there is an isometric cycle of length k in G going through u and v then the distance between (u, v) and (v, u) is $k/2$ in G_k.*

Proof. Assume G has an isometric cycle of length k, $C = \{c_1, c_2, \ldots c_{k/2}, c_{k/2+1}, \ldots c_k\}$ with $c_1 = u$ and $c_{k/2+1} = v$. As $d_G(u, v) = k/2$ and C is isometric we know that $c_{k/2+1}$ then must be v. We now know that $(c_1, c_{k/2+1})$, $(c_2, c_{k/2+2})$, ... , $(c_{k/2}, c_k)$ and $(c_{k/2+1}, c_1)$ are vertices in G_k, and there clearly is an edge between each consecutive pair of these vertices. Thus $d_{G_k}[(u, v),(v, u))] \leq k/2$. By Lemma 3.1 we have that $d_{G_k}[(u, v),(v, u)] \geq k/2$ which ensures equality and completes the proof. ∎

Lemma 3.3 *If k is even and the distance between (u, v) and (v, u) is $k/2$ in G_k then there is an isometric cycle of length k in G going through u and v.*

Proof. Let $d_{G_k}[(u, v),(v, u)] = k/2$ and let $P = \{(u, v),(a_2, b_2),\ (a_3, b_3) \ldots (a_{k/2-1},\ b_{k/2-1}),\ (v, u)\}$ be a shortest path between (u, v) and (v, u). Now, obviously $W = \{u,\ a_2,\ a_3 \ldots\ v,\ b_2,\ b_3 \ldots,\ u\}$ is a cyclic walk of length k. By the definition of a walk, W is also a subgraph in G. In order to obtain a contradiction let us assume that there is a pair of vertices a and b in W with $d_G(a, b) < d_W(a, b)$. Now, let x be a vertex in W so that either (a, x) or (x, a) is in P. As a and x are on opposite sides of the cyclic walk W, there is a walk of length $k/2$ from a to x going through b. This means that $d_W(a, b) + d_W(b, x) \leq k/2$. But $d_G(a, b) < d_W(a, b)$ implying $d_G(a, x) \leq d_G(a, b) + d_G(b, x) < d_W(a, b) + d_W(b, x) \leq k/2$ and contradicting that $d_G(a, x) = k/2$. We can now conclude that $d_G(a, b) = d_W(a, b)$ for every a and b in $V(W)$, meaning that W is an isometric cycle of length k. ∎

Together the lemmas above yield an equivalence.

Corollary 3.4 *If k is even, there is an isometric cycle of length k in G if and only if there is a pair of vertices, u and v, with $d_{G_k}[(u, v),(v, u)] = k/2$.*

Now we have an analogous result for odd k. For a vertex (u, v) in G_k, we define the set $M'_k(u, v)$ to be the set $\{(u, x) : (u, x) \in V(G_k) \wedge (v, x) \in E(G)\}$. Using this set, we can show the following.

Lemma 3.5 *If k is odd and there is an isometric cycle of length k in G, going through u and v, the distance between (u, v) and a vertex (v, x) in $M'_k(v, u)$ is $\lfloor k/2 \rfloor$ in G_k.*

Proof. Assume G has an isometric cycle of length k, $C = \{c_1, c_2, ...c_{\lfloor k/2 \rfloor}, c_{\lfloor k/2 \rfloor+1}, ...c_k\}$ with $c_1 = u$, $c_{\lfloor k/2 \rfloor+1} = v$ and $c_k = x$. As in the proof of Lemma 3.3, $(c_1, c_{\lfloor k/2 \rfloor+1})$, $(c_2, c_{\lfloor k/2 \rfloor+2})$, ... , $(c_{\lfloor k/2 \rfloor}, c_{k-1})$ and $(c_{\lfloor k/2 \rfloor+1}, c_k)$ are vertices in G_k. Again we can see that there is an edge between each consecutive pair of vertices, and that $(c_{\lfloor k/2 \rfloor+1}, c_k)$ must be in $M'_k(v, u)$, because $c_{\lfloor k/2 \rfloor+1}$ actually is v and there is an edge between c_k and c_1 in G. The given path assures us that $d_{G_k}[(u, v), (v, x)] \leq \lfloor k/2 \rfloor$ while Lemma 3.1 gives $d_{G_k}[(u, v), (v, x)] \geq \lfloor k/2 \rfloor$ and ensures equality. ∎

Lemma 3.6 *If k is odd and the distance in G_k between (u, v) and a vertex (v, x) in $M'_k(v, u)$ is $\lfloor k/2 \rfloor$, then there is an isometric cycle of length k in G, going through u and v.*

Proof. Let $d_{G_k}[(u, v), (v, x)] = \lfloor k/2 \rfloor$ and let $P = \{(u, v), (a_2, b_2), (a_3, b_3) \ldots (a_{\lfloor k/2 \rfloor}, b_{\lfloor k/2 \rfloor}), (v, x)\}$ be a shortest path between (u, v) and (v, x). We observe that $W = \{u, a_2, a_3 \ldots v, b_2, b_3 \ldots, x, u\}$ must be a cyclic walk of length k and a subgraph in G. Assume for a contradiction that there is a pair of vertices a and b in W with $d_G(a, b) < d_W(a, b)$. Let z be a vertex in W so that either (a, z) or (z, a) is in P and $d_W(a, b) + d_W(b, z) \leq \lfloor k/2 \rfloor$. We can always find such a vertex by starting at a and walking $\lfloor k/2 \rfloor$ steps along W either clockwise or counterclockwise. The direction to walk in is the one that makes us visit b on the way to z. Since we assumed that $d_G(a, b) < d_W(a, b)$ it follows that $d_G(a, z) \leq d_G(a, b) + d_G(b, z) < d_W(a, b) + d_W(b, z) \leq \lfloor k/2 \rfloor$ which obviously contradicts to $d_G(a, x) = \lfloor k/2 \rfloor$. This can only mean that W is an isometric cycle. ∎

Corollary 3.7 *If k is odd, there is an isometric cycle of length k in G if and only if there are vertices u, v and x so that $x \in M'_k(v, u)$ and $d_{G_k}[(u, v), (v, x)] = \lfloor k/2 \rfloor$.*

In order to simplify notation, and to be able to treat k odd and k even as one case, for a $(u, v) \in V(G_k)$ we define the set $M_k(u, v)$ to be (u, v) if k is even and $M'_k(u, v)$ if k is odd. Now we are able to summarize the above section in one result.

Theorem 3.8 *G has an isometric cycle of length k if and only if there are vertices u and v and $x \in V(G)$ so that $(v, x) \in M_k(v, u)$ and $d_{G_k}[(u, v), (v, x)] = \lfloor k/2 \rfloor$.*

Proof. If k is even the result is equivalent to Corollary 3.4, in the odd case it is the same as Corollary 3.7. ∎

4 Algorithm and complexity analysis

It should be clear how one can use G_k to check whether G has an isometric cycle of length k. We use a straightforward approach - For a given k, compute G_k, and find out whether there is a pair of vertices (u, v) and (v, x) in $V(G_k)$ so that $(v, x) \in M_k(v, u)$ and $d_{G_k}[(u, v), (v, x)] = \lfloor k/2 \rfloor$. If such a pair exists we have a cycle of length k, if not - we don't. Now we do this for every k between 3 and n, and this way we find the longest isometric cycle. We observe that Lemma 3.1 guarantees that $d_{G_k}[(u, v), (v, x)] \geq \lfloor k/2 \rfloor$, so we can search for vertices satisfying the inequality $d_{G_k}[(u, v), (v, x)] \leq \lfloor k/2 \rfloor$ instead of the equation above.

Theorem 4.2 *Given a graph G, Algorithm LIC computes the length of the longest isometric cycle of G*

Proof. If G is a tree it has no cycles and the algorithm correctly returns 0. If it has a cycle, it must have an isometric cycle of length at least 3 and at most n. Assume the longest isometric cycle of G has length k'. Then Theorem 3.8 states that G_k must have a pair of vertices (u, v) and (v, x) so that $(v, x) \in M(v, u)$ and $d_{G_k}[(u, v), (v, x)] = \lfloor k/2 \rfloor$. This means that $[(u, v), (v, x)]$ must be in $E_k^{\lfloor k/2 \rfloor}$ so the variable ans will be set to k' in the iteration of the outer loop that has $k = k'$. For all iterations after this, e.g. with $k > k'$, the same theorem states that there cannot be any vertices (u, v) and (v, x) that satisfy the above conditions. This ensures that the command $ans := k$ will not be executed. Thus the algorithm terminates with the value of ans equal to k' - the length of Gs longest isometric cycle. ∎

Now that we have proven the correctness of Algorithm LIC we can move on to analyzing its time complexity and discussing some of the implementation details. As the algorithm is fairly straightforward, the complexity analysis also is quite simple. We can compute the distance matrix and find out whether G is acyclic using naive algorithms in $\mathcal{O}(n^3)$ time. Having pre-computed the distance matrix of G, we now can make queries about the

Algorithm 4.1
LIC - Longest Isometric Cycle
Input: A graph $G = (V, E)$.
Output: The length of the longest isometric cycle in G - ans
begin
    $ans := 0$
    Compute the distance matrix of G.
    if G is a tree **then**
        return ans
    end-if
    for every k from 3 **to** n **do**
        $V_k := \emptyset$
        for every u and v in V **do**
            if $d(u, v) = \lfloor k/2 \rfloor$ **then**
                $V_k := V_k \cup \{(u, v)\}$
            end-if
        $E_k := \emptyset$
        for every (u, v) and (w, x) in V_k **do**
            if $(u, w) \in E \wedge (v, x) \in E$ **then**
                $E_k := E_k \cup \{[(u, v), (w, x)]\}$
            end-if
        $G_k := (V_k, E_k)$
        Compute $G_k^{\lfloor k/2 \rfloor} = (V_k, E_k^{\lfloor k/2 \rfloor})$
        for every triple of vertices (u, v, x) in V **do**
            if $(u, v) \in V(G_k) \wedge (v, x) \in M_k(v, u) \wedge [(u, v), (v, x)] \in E_k^{\lfloor k/2 \rfloor}$ **then**
                $ans := k$;
            end-if
    return ans
end

Figure 1: The algorithm for computing the longest isometric cycle.

distance between two vertices in G in constant time. Using this we see that V_k is computed in $\mathcal{O}(n^2)$ time for a given k.

Now we observe that E_k is computed in $|V_k|^2$ time. At this point, we arrive at the spot where we have to compute $G_k^{\lfloor k/2 \rfloor}$. The fastest known way to do that is to use the folklore algorithm for computing graph powers; To compute G^x we write x in base 2 and let d_{i+1} be the i'th digit in this string counting from right to left. Now we find G^{2^k} for $2^k \leq x$ and compute the matrix product $\prod_{i=0}^{\lfloor log(x) \rfloor} A_i$ where A_i is defined to be G^{2^i} if $d_i = 1$ and the identity matrix otherwise. It is easy to show that the time complexity of this approach is $|V(G)|^\alpha \log(x)$ when n^α is the time needed to multiply two n by n matrices. This means that computing $G_k^{\lfloor k/2 \rfloor}$ from G_k in this manner takes $\mathcal{O}(|V_k|^\alpha \log(\lfloor k/2 \rfloor))$ time.

Having computed $G_k^{\lfloor k/2 \rfloor}$, we only have the last loop left. The loop iterates over all triples (u, v, x) of vertices in V. Having pre-computed G_k, $G_k^{\lfloor k/2 \rfloor}$ and the distance matrix of G, we can perform all the tests in the following if-sentence in $\mathcal{O}(1)$ time. Note that we do not actually compute the set $M_k(v, u)$, we only test whether (v, x) satisfies the conditions to be in the set. To conclude the analysis we just need to summarize the discussion above and make a couple observations.

Observation 4.3 $\sum_{k=3}^{n} |V_k| \leq 2n^2$

Proof. For given k_1 and k_2 with $\lfloor k_1/2 \rfloor \neq \lfloor k_2/2 \rfloor$ we see that V_{k_1} and V_{k_2} are pairwise disjoint subsets of V^2. If $\lfloor k_1/2 \rfloor = \lfloor k_2/2 \rfloor$ then $V_{k_1} = V_{k_2}$ by the definition of G_k. By summing over all even and all odd k's we obtain $\sum_{k=3}^{n} |V_k| = \sum_{k=1}^{\lfloor (n-1)/2 \rfloor} |V_{2k+1}| + \sum_{k=2}^{\lceil n/2 \rceil} |V_{2k}| \leq |V^2| + |V^2| = 2n^2$. ∎

Theorem 4.4 *If $\mathcal{O}(n^\alpha)$ is the time needed to multiply two n by n binary matrices and $\alpha \geq 2$, Algorithm LIC terminates in $\mathcal{O}(n^{2\alpha} \log(n))$ steps.*

Proof. Let T be the total number of steps performed by the algorithm. By the discussion above, we see that $T = \mathcal{O}(n^3) + \sum_{k=3}^{n} [\mathcal{O}(n^2) + \mathcal{O}(|V_k|^2) + \mathcal{O}(|V_k|^\alpha \log(\lfloor k/2 \rfloor)) + \mathcal{O}(n^3)]$ By rearranging our terms and summing the terms not dependant on k we obtain $T = \mathcal{O}(n^4) + \sum_{k=3}^{n} \mathcal{O}(|V_k|^\alpha \log(\lfloor k/2 \rfloor))$. Now $\log(\lfloor k/2 \rfloor) = \mathcal{O}(\log(n))$ so we end up with $T = \mathcal{O}(n^4) + \mathcal{O}(\log(n)) \sum_{k=3}^{n} \mathcal{O}(|V_k|^\alpha)$. As $\alpha \geq 2$, n^α is a convex function and we can put the summation inside the $\mathcal{O}$. This yields $T = \mathcal{O}(n^4) + \mathcal{O}(\log(n))\mathcal{O}([\sum_{k=3}^{n} |V_k|]^\alpha)$ while Observation 4.3 allows us to simplify the expression to $T = \mathcal{O}(n^4) +$

$\mathcal{O}(\log(n)(2n^2)^\alpha)$. As $2\alpha \geq 4$ we may simplify even further, to $T = \mathcal{O}(n^{2\alpha} \log(n))$. ▪

Finally, we recall the fastest known algorithm for matrix multiplication.

Theorem 4.5 *[1] Two n by n matrices can be multiplied in $\mathcal{O}(n^{2.376})$ time.*

Corollary 4.6 *Algorithm LIC runs in $\mathcal{O}(n^{4.752}\log(n))$ time.*

5 Conclusion

A way to view isometric subgraphs is as a generalization of induced subgraphs. While finding the longest induced cycle in a graph is hard [3], we have seen that finding the longest isometric cycle is easy. Finding the longest isometric path is trivial, as any isometric path must be a shortest path and vice versa. This might lead us to believe that finding out whether a graph G contains a given graph H as an isometric subgraph also might be solvable in polynomial time. Unfortunately, one can show that this is not the case unless $\mathcal{P} \neq \mathcal{NP}$, because the problem becomes $\mathcal{NP}$-complete even when H is restricted to the class of stars. A simple reduction from $IndependentSet$ works as follows. On input (G, k) to $IndependentSet$, construct G' by taking G and adding a new vertex u and making u adjacent to all vertices of G. It is now easy to see that G' has an isometric star on k edges if and only if G has an independent set of size k.

References

[1] Don Coppersmith, Shmuel Winograd. Matrix multiplication via arithmetic progressions. *Journal of Symbolic Computation*, v.9 n.3, p.251-280, March 1990.

[2] Richard M. Karp. Reducibility among Combinatorial Problems. *Complexity of Computer Computations*, Plenum Press, p.85-103,1972.

[3] M. R. Garey and D. S. Johnson. *Computers and Intractability.* W. H. Freeman and Co., 1978.

[4] Douglas B. West. *Introduction to Graph Theory, Second edition.* Prentice Hall, p.1, 2001.