Discrete visibility problems and graph algorithms

Enrico Puppo
Istituto per la Matematica Applicata
Consiglio Nazionale delle Ricerche
Via De Marini, 6 (Torre di Francia) - 16149 Genova - ITALY
Email: PUPPO@IMA.GE.CNR.IT

Paola Marzano
Dipartimento di Informatica e Scienze dell'Informazione
Università di Genova
Viale Benedetto XV, 4 - 16132 Genova - ITALY
Email: MARZANO@DISI.UNIGE.IT

Abstract

Many problems of practical interest involve line-of-sight on a topographic surface. Some such problems can be successfully studied on the basis of the mutual visibility among a finite number of representative points. Such visibility problems can be formalized and resolved as graph problems. In this paper, we show that graph algorithms can be useful to find efficient solutions for discrete visibility problems in several cases. On the basis of results from the theory of complexity, we give some practical rules to apply such an approach. We further investigate the solution of some relevant visibility problems under this perspective.

1 Introduction

Problems related to the visibility of points on a topographic surface have practical impact in many important applications. Several aspects of these problems have been faced by many researchers during the last twenty years, from both a practical and a theoretical point of view: for recent extensive surveys on this subject see (De Floriani et Magillo, 1994) and (Nagy, 1994).

Although many visibility problems only require knowledge about the viewshed for a given, fixed observation point, some optimization problems may require knowledge about the mutual visibility between arbitrary pairs of sites on the terrain. Examples of such problems are the placement of fire towers, communication relays, and transmission stations; the planning of paths that must either remain hidden from a set of observers, or be scenic in viewing as much landscape as possible; and the locus of travel of parties that must remain in constant line-of-sight communication. Dynamic scenarios, in which viewpoints and/or observed points are moving on the terrain surface, give the most complicated situations in this respect. All such problems are likely to be quite complicated, and algorithms to resolve them are likely to be both time and memory consuming. Indeed, not only do they involve a large amount of information about visibility, but also their inherent computational complexity is rather high. For this reason, most attempts to give them practical solutions are based on heuristics.

A possible way to reduce the input size of visibility problems is to restrict the sets of viewpoints, or observed points, or both, to some discrete locations distributed on the terrain surface. A model that encodes information about mutual visibility only for a finite set of representative points is called a *discrete visibility model*. The use of a discrete model, though limited to an approximate view of the terrain, is common practice in the context of most applications: for example, any model based on the pixel-to-pixel visibility in gridded elevation models is indeed a discrete visibility model. Line-of-sight problems treated in such a framework are called *discrete visibility problems*.

Discrete visibility problems can be formalized as graph problems on a structure called a *visibility graph*, encoding the mutual visibility between pairs of representative points. The huge existing literature on graph algorithms can be exploited to find effective solutions for discrete visibility problems. Some visibility problems exhibit very high computational complexity even under the discrete formulation: in order to solve such problems, approximate graph algorithms can be often applied successfully in practice.

A natural question in this respect is whether using general tools from the theory of graph algorithms is indeed a valid approach to the solution of discrete visibility problems. Being a mature field, with an enormous number of applications, the literature on graph algorithms generally offers tools that are effective and efficient not only in theory, but also in practice. On the other hand, one might argue that possibly different techniques could result in more efficient line-of-sight problems. In this paper we try to give an answer to this question both by setting some theoretical results, and by showing that graph algorithms can be effectively applied in resolving discrete visibility problems.

Given a (sufficiently complex) discrete visibility problem that can be formalized as a graph problem, theoretical results presented in §3 can be informally summarized as follows:

1. if an optimal solution is known for the graph problem, it is an optimal solution also for the visibility problem;
2. if the graph problem is intractable, also the visibility problem is intractable;
3. if approximate/suboptimal algorithms for the (intractable) graph problem are known, which give solutions within some bound from the optimal solution, such algorithms and bounds also apply to the visibility problem.

On this basis, we give some practical rules for handling visibility problems through graph algorithms, and in §4 we describe their application to some visibility problems of practical interest.

Preliminaries on terrain visibility and algorithm reduction are given in §2.1 and §2.2, respectively. In §5 we add some remarks on the practical use of discrete visibility techniques. Open problems and concluding remarks are finally outlined in §6.

2 Preliminaries

2.1 Terrain visibility

In this section we give a formal definition of terrain and terrain visibility. We review the main terrain models considered in the literature, and we consider computational issues about computing and storing visibility information for such models.

2.1.1 Terrain models

A *topographic surface* (or *terrain*) σ is the image of a real bivariate function f defined over a domain D in the Euclidean plane:

$$\sigma = \{(x, y, f(x, y)) \mid (x, y) \in D\}.$$

It should be noticed that this definition does not allow *overhangs*, which are rare, though sometimes present in natural terrains.

Two classes of topographic surfaces are of special interest from a computational point of view:

- A *polyhedral terrain* is the image of a continuous piecewise-linear function f. A polyhedral terrain can be described on a partition of the domain D into polygonal regions $\{R_1, \ldots, R_k\}$ such that the image of f over each region R_i ($i = 1, \ldots, k$) is a planar patch. The most commonly used polyhedral terrains are *Triangulated Irregular Networks (TINs)*, in which all regions are triangles.
- A *gridded elevation model* (often identified with the more general term of *Digital Elevation Model*) is usually defined as follows:
  - a domain D is subdivided into regular rectangular (or square) regions induced by a regular grid over D;
  - the topographic surface is the image of a function f that is either bilinear or constant over each region: in the bilinear case f is continuous, while in the constant case there are discontinuities along the borders between adjacent regions (this latter model is also called a *stepped model*).

The shape of regions and the functions used over each region are not the only difference between polyhedral and gridded models. In common practice, while polyhedral terrains are always manipulated consistently with respect to their definition, gridded models (especially stepped models) are more often treated as discrete models. A *discrete gridded model* is a gridded model in which each region is considered as an atomic entity: no distinction can be made between different points lying in the same region. Regions are usually called *pixels*, for analogy with digital images, and the properties of terrain (e.g., elevation, slope, etc.) are given by enumeration at all pixels. Hence, if $P = \{X_1, \ldots, X_k\}$ is the set of all centroids of pixels in a gridded model, the corresponding discrete model can be formally defined as a restriction of function f to finite set $P \subset D$.

2.1.2 Visibility models

Intuitively, given two points P and Q on a topographic surface σ, we say that Q is visible from P if an observer standing at P can see an object placed at Q. In the literature, there are different assumptions about the fact that the observer at P or the observed object at Q or both can stand at a certain height above terrain. For the purpose of this paper, let us assume that any observer on σ stands at a height h_v above the terrain, and any observed object stands at a height h_o above the terrain, where h_v and h_o are two fixed non-negative numbers. Scenarios in which the height of observers or objects can be variables in a visibility problem are out of the scope of this paper, and are left out of our discussion intentionally. Given a topographic surface σ defined by a function f over a domain D, the oriented *line-of-sight* for a pair of points $((x_v, y_v), (x_o, y_o)) \in D \times D$ is the straight-line segment joining points $(x_v, y_v, f(x_v, y_v) + h_v)$ and $(x_o, y_o, f(x_o, y_o) + h_o)$ of $\mathbb{R}^3$. A Boolean *visibility function* v_σ is defined on $D \times D$ as follows:

$$v_\sigma(X_v, X_o) = \begin{cases} \textbf{true} & \text{if no point of the line-of-sight of } (X_v, X_o) \text{ lies below } \sigma \\ \textbf{false} & \text{otherwise} \end{cases}$$

In the definition above, X_v is called a *viewpoint*, and X_o is called an *observed point*. A more robust and conservative definition of visibility would require that the line-of-sight does not touch the terrain except at its endpoints. On the other hand, in this case a pair of points lying on the same face of a polyhedral terrain could not be considered mutually visible for $h_v = h_o = 0$: as this would be against intuition and common practice, we have adopted a less restrictive definition.

If $v_\sigma(X_v, X_o) = \textbf{true}$ we informally say that X_v *sees* X_o, and X_o *is visible* from X_v.

In general, function v_σ is not symmetric, i.e., it is possible that a point sees another point without being visible from it in turn (see Figure 2d). If $h_v = h_o$ then v_σ is symmetric: in this case, if $v_\sigma(X_v, X_o) = \textbf{true}$ we say that X_v and X_o are *mutually visible*.

A *visibility model* for σ is a computational framework that permits to evaluate v_σ over either its whole domain, or a portion of it. In the following we outline three different classes of visibility models, which can can be suitable for different visibility problems.

A *continuous visibility model* for σ encodes v_σ over the whole domain $D \times D$. In principle, complete knowledge of v_σ would be necessary to give an exact solution to many important problems. Although continuous visibility models exist, they are inherently too expensive in terms of combinatorial complexity to be of practical interest (Gigus et al., 1991, Agarwal and Sharir, 1994).

A *semi-continuous visibility model* encodes v_σ over $S_v \times D$, where S_v is a finite set of viewpoints. The case in which v_σ is represented over $D \times S_o$, where S_o is a finite set of observed points, is completely analogous, and it is also called a semi-continuous visibility model. Such a model is suitable, and can be used in practice, for all applications in which possible viewpoints (or observed points) can be fixed a priori. In the literature, different techniques and data structures have been proposed to compute and to store efficiently a restriction of v_σ to any domain $\{X_v\} \times D$, i.e. a continuous model of visibility from a single viewpoint X_v, commonly known as the *viewshed* (see (De Floriani et Magillo, 1994) for a survey). A semi-continuous visibility model can be obtained as a collection of continuous viewsheds for all points of S_v.

When both sets of viewpoints S_v, and of observed points S_o are finite, a model describing v_σ on $S_v \times S_o$ is called a *discrete visibility model.* Note that the two sets S_v and S_o need not necessarily be disjoint, and can even be coincident. A discrete visibility model can be represented by a graph $G_v = (S, E_v)$, called the *visibility graph*, where $S = S_v \cup S_o$, and

$$E_v = \{(X_v, X_o) \in S_v \times S_o \mid v_\sigma(X_v, X_o) = \mathbf{true}\ \}.$$

The arcs of the visibility graph are directed if v_σ is non-symmetric; for $h_v = h_o$ arcs are not directed. In the rest of the paper, we will refer to set S, containing both viewpoints and observed points, as the set of *representative points.*

Note that discrete gridded terrain models admit only discrete visibility models, since their domain is always finite. Let P be the set of pixels of a discrete gridded model: a visibility model in which $S_v = S_o = P$ is a discrete approximation of a continuous visibility model; a visibility model in which $S_v \subset P$ and $S_o = P$ is a discrete approximation of a semi-continuous visibility model. This remark implies that whenever a visibility problem is studied in the framework of a discrete terrain model - which is often the case in common practice - its formulation is intrinsically discrete.

2.1.3 Computational issues

A number of efficient algorithms and data structures have been developed in the literature for computing and storing visibility models. Most literature deals with either continuous viewsheds or discrete visibility. In general, the case of polyhedral terrains is more complicated than the case of gridded models, and it requires sophisticated computational techniques and data structures.

Here, we concentrate on discrete visibility models. In §2.2 formal definitions of time complexities and related notation are given. We evaluate the complexities in terms of three parameters:

- the number m of parameters needed for describing terrain σ;
- the number n_v of viewpoints in S_v;
- the number n_o of viewpoints in S_o.

For the case of polyhedral terrains, m is the number of vertices in the subdivision of D into polygonal regions: it is known from the Euler formula that the number of edges and regions in the subdivision are both in $O(m)$. For the case of gridded models, m is the number of regions (pixels) in the grid.

A discrete visibility model can be easily encoded through the adjacency matrix of the visibility graph G_v, called the *visibility matrix.* The visibility matrix is a matrix of $n_v \times n_o$ Boolean values, hence involving an $O(n_v n_o)$ storage cost. The visibility matrix can be built by computing the visibility for any possible pair of points $(X_v, X_o) \in S_v \times S_o$.

In the case of polyhedral terrains G_v can be computed trivially in $O(n_v n_o m)$ with a brute force algorithm that tests all $n_v n_o$ possible lines-of-sight between pairs of points of $S_v \times S_o$ against all $O(m)$ edges of σ that can possibly interrupt a line-of-sight. The projection of each line-of-sight on the domain D is considered, and all edges of the subdivision are tested for intersection. A line-of-sight is interrupted if and only if the terrain surface lies above it at least at an intersection point. A practical speedup technique consists in tracing the projected line-of-sight through the triangulation: starting at the first triangle intersected in the proximity of an endpoint, the line is traced by moving through adjacent triangles to the other endpoint. In this way, the cost of tracing is equal to the number of edges intersected, plus the location of the first triangle. Athough the performance does not improve in the worst case, in practice the average cost per line-of-sight will be of $O(\sqrt{m})$.

This time complexity in the worst case can be reduced to $O(n_v n_o \log m + n_v m\alpha(m) \log m)$ time, where $\alpha()$ is the extremely slowly growing functional inverse of Ackermann's function. Such a time complexity is achieved by using a sophisticated technique for ray shooting described in (Cole et Sharir, 1989). Indeed, for each viewpoint X_v it is possible to build a structure in $O(m\alpha(m) \log m)$ time that permits to test the visibility from X_v to any arbitrary point in $O(\log m)$ time: hence, all lines-of-sight can be tested in $O(n_v n_o \log m)$ time after computing n_v such structures.

In the case of gridded elevation models computing visibility is an easier task, because the regularity of the underlying grid can be exploited. The straight-line segment joining a viewpoint X_v to an observed point X_o can intersect at most $O(\sqrt{m})$ edges, which are easily found by traversing the grid. Hence, G_v can be computed in $O(n_v n_o \sqrt{m})$ time by the brute force algorithm outlined above. In principle, it is possible to apply the ray shooting technique mentioned above also to gridded models, by achieving the same asymptotic time complexity. However, this is to be considered more a theoretical than a practical possibility, since simple algorithms that can exploit the regularity of the grid usually beat in practice sophisticated computational geometry algorithms.

Note that in the case of discrete gridded models X_v and X_o are actually identified with the pixels containing them: the visibility is tested by scanning the digital straight-line segment that joins such pixels, and by testing the elevation of the line-of-sight against the elevation of terrain at each pixel of such a digital line. The digital line segment can involve at most $O(\sqrt{m})$ pixels, thus yielding the same time complexity as above.

Since discrete visibility models built over discrete gridded terrain models are often used as approximations of continuous visibility models, we also outline results that follow directly from the above analysis in the special cases when S_v and/or S_o coincide with the whole set of pixels P:

- the *discrete viewshed* (i.e., a visibility model on $\{X_v\} \times P$) can be computed in time $O(m^{\frac{3}{2}})$;
- the *complete discrete visibility model* (i.e., a visibility model on $P \times P$) can be computed in time $O(m^{\frac{5}{2}})$.

An approximation of a discrete viewshed can also be obtained in $O(m)$ by computing the running maximum of the tangent of the line-of-sight between the viewpoint and each of the

$O(\sqrt{m})$ pixels lying on the border of the gridded model: a single scanning of the projection of each line-of-sight on D is sufficient to determine the visibility of each pixel along it. This result on time complexity follows from the fact that any pixel in the grid is intersected by some such line. By using the same approach, an approximation of the complete visibility model is obtained in time $O(m^2)$. See (Teng, 1993) for details on this technique.

In the rest of the paper we will deal with discrete visibility models while remaining generic about the terrain model on which they are computed. However, we will assume that it is always possible to compute G_v in a time polynomial in m, n_v, and n_o.

2.2 Basic notions of algorithm complexity

We briefly review some basic notions necessary to the rest of the paper. Further details on this subject can be found in (Garey et Johnson, 1979, Preparata et Shamos, 1985, Cormen et al., 1990).

The *time complexity* $f(n)$ of an algorithm $\mathbf{A}$ is the amount of time spent by the algorithm to get to its end, where f is a function and n is the size of the input of $\mathbf{A}$, namely the cardinality of the input set. The *asymptotic time complexity* $g(n)$ is a function asymptotically proportional to $f(n)$, namely such that $\lim_{n\to\infty} \frac{f(n)}{g(n)} = c$, for some $c \in \mathbb{R}$ such that $c \neq 0$. The following notations are used to allow for rigorous statements:

- $O(f(n)) = \{g(n) \mid \exists\ c, n_0\ :\ g(n) \leq cf(n), \forall n \geq n_0\}$
- $\Omega(f(n)) = \{g(n) \mid f(n) \in O(g(n))\}$
- $\theta(f(n) = O(f(n)) \cap \Omega(f(n))$
- $o(f(n)) = \{g(n) \mid \forall c,\ \exists\ n_0\ :\ cg(n) < f(n), \forall n \geq n_0\}$
- $\omega(f(n)) = \{g(n) \mid f(n) \in o(g(n))\}$

2.2.1 Reduction and equivalence of algorithms

Informally, a generic problem $\mathcal{A}$ reduces to another problem $\mathcal{B}$ (denoted $\mathcal{A} \propto \mathcal{B}$) if it is possible to find a solution to $\mathcal{A}$ by resolving $\mathcal{B}$. More formally, the reduction can be characterized through the existence of two algorithms $\mathbf{I}_{\mathcal{A}\to\mathcal{B}}$ and $\mathbf{O}_{\mathcal{B}\to\mathcal{A}}$ of input and output conversion, respecively, satisfying the following requirements:

1. Given an input $I_{\mathcal{A}}$ for problem $\mathcal{A}$, algorithm $\mathbf{I}_{\mathcal{A}\to\mathcal{B}}$ converts $I_{\mathcal{A}}$ into an input $I_{\mathcal{B}}$ for problem $\mathcal{B}$.
2. If $O_{\mathcal{B}}$ is the solution of problem $\mathcal{B}$ for input $I_{\mathcal{B}}$ defined as above, algorithm $\mathbf{O}_{\mathcal{B}\to\mathcal{A}}$ converts $O_{\mathcal{B}}$ into the solution $O_{\mathcal{A}}$ of problem $\mathcal{A}$ for input $I_{\mathcal{A}}$.

These requirements are summarized in the commutative diagram depicted in Figure 1, where symbols $\mathbf{A}_{\mathcal{A}}$ and $\mathbf{A}_{\mathcal{B}}$ denote two virtual algorithms for resolving problems $\mathcal{A}$ and $\mathcal{B}$, respectively, and arrows denote the application of algorithms.

$$\begin{array}{ccc} I_{\mathcal{A}} & \xrightarrow{\mathbf{I}_{\mathcal{A}\to\mathcal{B}}} & I_{\mathcal{B}} \\ \mathbf{A}_{\mathcal{A}} \downarrow & & \downarrow \mathbf{A}_{\mathcal{B}} \\ O_{\mathcal{A}} & \xleftarrow{\mathbf{O}_{\mathcal{B}\to\mathcal{A}}} & O_{\mathcal{B}} \end{array}$$

Figure 1: Problem $\mathcal{A}$ reduces to problem $\mathcal{B}$ if the diagram is commutative.

The *computational cost* of reduction is the sum of the worst-case time complexities of algorithms $\mathbf{I}_{\mathcal{A}\to\mathcal{B}}$ and $\mathbf{O}_{\mathcal{B}\to\mathcal{A}}$. We will denote the fact that problem $\mathcal{A}$ can be reduced to problem $\mathcal{B}$ at cost T by $\mathcal{A} \propto_T \mathcal{B}$.

Reduction is common practice in complexity theory, with two main purposes:

- *upper bound via reduction*: if problem $\mathcal{A}$ can be reduced to $\mathcal{B}$ at a cost $T_{\mathcal{A}\to\mathcal{B}}$, and if problem $\mathcal{B}$ can be resolved at a cost $T_{\mathcal{B}}$, then problem $\mathcal{A}$ can be resolved in at most $T_{\mathcal{A}\to\mathcal{B}} + T_{\mathcal{B}}$ time.
- *lower bound via reduction*: if problem $\mathcal{A}$ has a lower bound $T_{\mathcal{A}}$, and if $\mathcal{A}$ can be reduced to $\mathcal{B}$ at a cost $T_{\mathcal{A}\to\mathcal{B}}$, then problem $\mathcal{B}$ has a lower bound of at least $T_{\mathcal{A}} - T_{\mathcal{A}\to\mathcal{B}}$.

If $\mathcal{A} \propto \mathcal{B}$ and $\mathcal{B} \propto \mathcal{A}$, and the costs of both reductions are smaller than the computational complexity of any of the two problems, then the computational complexities of $\mathcal{A}$ and $\mathcal{B}$ are coincident, and the two problems are called *equivalent*. This means that an efficient algorithm resolving one of them can be used to resolve efficiently the other one (through input/output conversion).

2.2.2 NP-complete problems and approximate solutions

An important class of problems that are computationally intractable in practice is the class of *NP-complete* problems. The theory of NP-completeness is fairly complicated, and the general question on whether NP-complete problems can be solved in polynomial time is one among the most famous open problems in theoretical computer science. It is not our purpose here to give thorough and formal notions about this theory ((Garey et Johnson, 1979) is an excellent reference for this). For practical purposes, the NP-completeness of a problem implies that there is no hope of finding an algorithm able to solve it in reasonable time for any dataset of relevant size.

A classical method for proving that a problem $\mathcal{B}$ is NP-complete consists in finding another problem $\mathcal{A}$ that is known to be NP-complete, and such that $\mathcal{A}$ can be reduced to $\mathcal{B}$ at a cost that is polynomial in its input size. Actually, algorithm reduction only proves the *NP-hardness* of $\mathcal{B}$. In order to prove its NP-completeness it is also necessary to show that any guessed solution for $\mathcal{B}$ can be tested to be a true solution in polynomial time. This part of the proof is usually straightforward, and has no relevance for the purposes of this paper. We omit these details in order to keep the theoretical background needed to read the paper as light as possible. Nevertheless, we insist in speaking of NP-completeness rather than NP-hardness because this is indeed the case in the examples we discuss in §4.

In spite of their intractability, NP-complete problems arise quite often in practical applications. A practical approach to their solution is based on heuristics. Some heuristic algorithms for NP-complete problems provide an exact solution, but they cannot warrant any time bound: they are just designed using "clever" strategies that "hopefully" can find a solution in a short time. Some other heuristic algorithms admit a reasonable upper bound in their running time, but they do not necessarily provide a true solution.

Quite often NP-complete problems involve optimization, i.e., they ask to minimize some cost under given constraints: for such problems it is sometimes possible to find suboptimal algorithms that warrant a solution satisfying the constraints, and whose cost is within some known bound from the cost of the optimal solution. Whenever such algorithms are available, they should be preferred to heuristics because they can give better guarantees on the quality of their results.

3 Reducing discrete visibility problems to graph problems

Some visibility problems can be studied and resolved in a simplified context that does not require knowledge of a complete visibility model. Here, we rather deal with visibility problems that are likely to be characterized by a high computational complexity (e.g., problems involving some optimization in placing objects or moving on the terrain). In the context of this section, we will remain generic about the visibility problem considered, by simply requiring that it must be complex enough, in its formulation and solution, to need information encoded in the visibility graph. We will investigate how the complexities of such kind of problems and of their corresponding graph problems may be related, and how such relations can be exploited to find efficient solutions to discrete visibility problems.

Let us call a *combinatorial structure* of the elements of a set of points S any structure that is finitely describable on the sole basis of the points of S considered as set elements, while disregarding their geometric structure (e.g., a subset of S, an ordering of elements of S, a graph having the points of S as nodes, etc.). A generic discrete visibility problem $\mathcal{V}$ is characterized by the following requirements:

v1. The input of $\mathcal{V}$ is expressed in terms of a terrain σ, described by m parameters, a set S_v of n_v viewpoints, and a set S_o of n_o observed points on σ, plus, possibly, any combinatorial structure based on the elements of $S = S_v \cup S_o$. The cardinality of S is denoted by n, where $\max(n_o, n_v) \leq n \leq n_o + n_v$.

v2. The output of $\mathcal{V}$ is a combinatorial structure of points of S.

A generic graph problem $\mathcal{G}$ is characterized by the following requirements:

g1. The input of $\mathcal{G}$ is described by a (possibly directed) graph G having n nodes, plus, possibly, any combinatorial structure based on the nodes of G.

g2. The output of $\mathcal{G}$ is a combinatorial structure of nodes of G.

We will often consider graph problems in their directed version. Results on the complexity of the same problems in their undirected version can be always used, as stated in the following remark, whose proof is straightforward:

Remark 3.1 *Let $\mathcal{G}_u$ be a graph problem defined for undirected graphs and let $\mathcal{G}_d$ be a version of the same problem for directed graphs. Problem $\mathcal{G}_u$ always reduces to $\mathcal{G}_d$ in a time linear in the input size of $\mathcal{G}_u$.*

Polynomial (linear) reduction between $\mathcal{G}_u$ and $\mathcal{G}_d$ is useful whenever a proof of NP-completeness is needed for $\mathcal{G}_d$. Indeed, if NP-completeness has been proved for $\mathcal{G}_u$, it holds also for $\mathcal{G}_d$.

Let $\mathcal{V}$ be a generic discrete visibility problem. Requirements v1 and v2 imply that $\mathcal{V}$ can be always reduced to a problem $\mathcal{G}_\mathcal{V}$ on the visibility graph G_v. Reduction of $\mathcal{V}$ to $\mathcal{G}_\mathcal{V}$ is straightforward, since the input conversion algorithm will just consist of the computation of G_v, while $\mathcal{G}_\mathcal{V}$ can be defined in such a way that its solution will be coincident with the solution of $\mathcal{V}$, thus making the output conversion trivial. Since such a reduction is so simple, we will call it a *translation* of $\mathcal{V}$ into $\mathcal{G}_\mathcal{V}$. Despite its conceptual simplicity, note that translation implies the cost of computing G_v, which can be quite expensive. This fact, suggests that we should use translation only for visibility problems that are sufficiently complex, i.e., that may need complete information about the discrete visibility model in order to be resolved.

Can we use knowledge about $\mathcal{G}_\mathcal{V}$ to state anything about the computational complexity of $\mathcal{V}$? In the following, we give results that help setting a lower bound for $\mathcal{V}$ on the basis of $\mathcal{G}_\mathcal{V}$, provided that the complexity of $\mathcal{G}_\mathcal{V}$ is not too small. We consider here the case of non-symmetric visibility, i.e., when the visibility graph is directed. It is straightforward to derive the same results for the symmetric case.

Lemma 3.2 *Given any directed graph $G = (N, A)$ with n nodes, there exists an embedding S of the nodes of N on a terrain σ, such that the visibility graph G_v for σ restricted to $S \times S$ is isomorphic to G. Moreover, S and σ can be found in time $O(n^2)$.*

Proof: The embedding S and the surface σ can be built by first placing the nodes of N at the vertices of a regular polygon with n sides inscribed into a circle with radius one, and then by placing suitable cones on the flat plane, whose purpose is to prevent visibility by interrupting a line-of-sight. Each cone has a base circle of radius ε_n, depending on the number n of vertices.

Let h_v and h_o be the fixed height of viewpoints and observed points, respectively, as defined in §2.1.2, with $h_v > h_o$. Initially, all pairs of vertices are mutually visible (see Figure 2b). In order to obtain the desired visibility graph, we now interrupt a line-of-sight from a point P to a point Q, whenever it does not correspond to an arc of A, through a cone of height h_n placed near Q (see Figure 2c).

It is easy to find, by elementary geometry, a suitable value of ε_n, a suitable location for the obstructing cone, and a suitable height h_n for the cone in such a way that no line-of-sight except $\vec{PQ}$ is interrupted, and that different cones do not overlap. Namely, the position

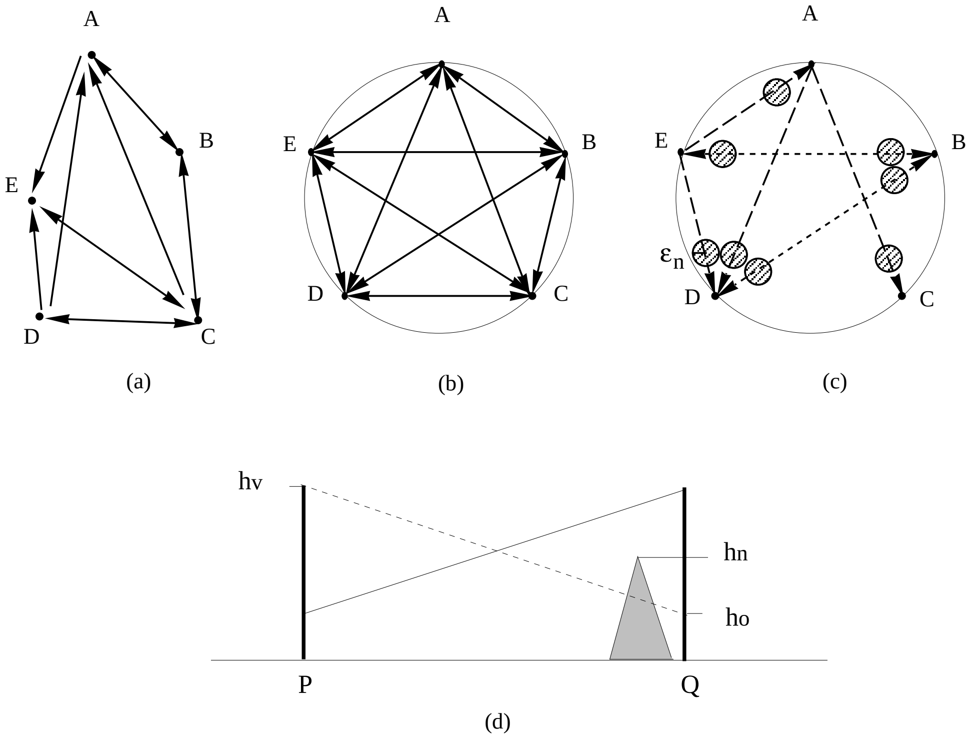

Figure 2: A graph G (a); complete visibility on the polygon (b); obstruction of lines-of-sight with cones (c); asymmetric obstruction of a line-of-sight (d).

and radius of each cone ensure that different cones do not intersect each other, and that no line-of-sight can be intersected by the cone, except possibly $\vec{PQ}$ and $\vec{QP}$. The suitable height h_n ensures that only $\vec{PQ}$ will be interrupted (see Figure 2d).

Detailed computations (omitted here for brevity) depend on the angle between two consecutive edges incident on a vertex, on the distance of such vertex from the first edge intersected by its incident edges, and on the relations between the heights of viewpoints and observed points.

Note that a similar construction can be made even if we require that σ is a polyhedral terrain, by substituting cones with suitable pyramids.

In summary, the embedding S is obtained in time $O(n)$, and a maximum of $O(n^2)$ edges of the complete graph are interrupted in time $O(1)$ each, for a total complexity of $O(n^2)$ in the worst-case. □

In practice, Lemma 3.2 states that the class of visibility graphs is not a special class of graphs. This means that any graph problem, whose input is a visibility graph G_v, maintains the same computational complexity as the standard graph problem on a generic input graph. On the basis of Lemma 3.2, we can state the following

Theorem 3.3 *Any graph problem $\mathcal{G}$, having an input graph with n nodes, can be reduced to a discrete visibility problem $\mathcal{V}_{\mathcal{G}}$ in $O(n^2)$. In particular, if a visibility problem $\mathcal{V}$ translates into graph problem $\mathcal{G}_{\mathcal{V}}$, then also $\mathcal{G}_{\mathcal{V}} \propto_{n^2} \mathcal{V}$.*

Proof: Given a directed graph G with n nodes, we know from Lemma 3.2 that we can obtain an embedding of nodes of G into a set of representative points S on a terrain σ in $O(n^2)$ time, such that G is the visibility graph for σ restricted to $S \times S$. Hence, $\mathcal{V}_{\mathcal{G}}$ is defined by translating "graph" into "terrain", "node" into "representative point", and "arc" into "line-of-sight". The output of $\mathcal{V}_{\mathcal{G}}$ will coincide with the output of $\mathcal{G}$ (modulo translation). In particular, if a problem $\mathcal{G}_{\mathcal{V}}$ has been obtained through translation from a visibility problem $\mathcal{V}$, the above reduction will be symmetric to translation, returning a problem equivalent to $\mathcal{V}$ for the visibility among the representative points of S. $\square$

In practice, Theorem 3.3 states that translation between discrete visibility problems and graph problems is bi-directional, though at different costs: translation from visibility problem to graph problem requires evaluating the visibility graph, while the reverse translation requires evaluating the embedding of the graph. An immediate consequence of this fact is the following

Corollary 3.4 *Let $\mathcal{V}$ be a discrete visibility problem, and let n be the number of representative points in the input of $\mathcal{V}$. If $\mathcal{V}$ translates into a graph problem $\mathcal{G}_{\mathcal{V}}$, and the lower bound asymptotic time complexity of $\mathcal{G}_{\mathcal{V}}$ is $T_{\mathcal{G}_{\mathcal{V}}}(n) \in \omega(n^2)$, then the lower bound $T_{\mathcal{V}}(n)$ for $\mathcal{V}$ cannot be asymptotically smaller than $T_{\mathcal{G}_{\mathcal{V}}}(n)$.*

Proof: Admit that a solution can be found for $\mathcal{V}$ in $T_{\mathcal{V}}(n) \in o(T_{\mathcal{G}_{\mathcal{V}}}(n))$. Then, we could find a solution to $\mathcal{G}_{\mathcal{V}}$ through its reduction to $\mathcal{V}$, in time $T_{\mathcal{V}}(n) + O(n^2) \in o(T_{\mathcal{G}_{\mathcal{V}}}(n))$. This contradicts our hypothesis that $T_{\mathcal{G}_{\mathcal{V}}}(n)$ is a lower bound fo $\mathcal{G}$. $\square$

From Corollary 3.4, we derive the following practical rule:

Rule 3.5 *If we are facing a discrete visibility problem $\mathcal{V}$, we can translate it into a graph problem $\mathcal{G}_{\mathcal{V}}$, and look in the literature for such a graph problem. If a lower bound is known for $\mathcal{G}_{\mathcal{V}}$, we know what we can expect in resolving $\mathcal{V}$, in terms of computational complexity. In the worst case, we could find that $\mathcal{G}_{\mathcal{V}}$ is an intractable problem: hence, we would be given a valid argument for using either approximate or heuristic techniques to resolve $\mathcal{V}$.*

Note that Corollary 3.4 does not apply if the lower bound of $\mathcal{G}_{\mathcal{V}}$ is $O(n^2)$. This means that if $\mathcal{V}$ translates into a problem $\mathcal{G}_{\mathcal{V}}$ whose computational complexity is rather simple, it is possible that the computational complexity of $\mathcal{V}$ is even simpler than the complexity of $\mathcal{G}_{\mathcal{V}}$. This fact is consistent with our initial assumptions that $\mathcal{V}$ must be complex enough to require knowledge of the discrete visibility model, whose spatial complexity can be as large as $O(n^2)$.

Assume now that we are given $\mathcal{V}$ and $\mathcal{G}_{\mathcal{V}}$ as above, and that an algorithm for resolving $\mathcal{G}_{\mathcal{V}}$ is known. $\mathcal{V}$ can be resolved through the following steps:

1. compute the visibility graph G_v;
2. resolve $\mathcal{G}_{\mathcal{V}}$ on G_v.

Is this an efficient way to resolve $\mathcal{V}$? An answer to this question is given by the following

Theorem 3.6 *If the lower bound $T_{\mathcal{G}_\mathcal{V}}(n)$ of $\mathcal{G}_\mathcal{V}$ is asymptotically greater than quadratic and asymptotically greater than the complexity of computing G_v (from σ and S), and an algorithm $\mathbf{A}_{\mathcal{G}_\mathcal{V}}$ resolves $\mathcal{G}_\mathcal{V}$ in optimal time $\theta(T_{\mathcal{G}_\mathcal{V}}(n))$, then $\mathbf{A}_{\mathcal{G}_\mathcal{V}}$ is also optimal for resolving $\mathcal{V}$ (through the procedure described above).*

Proof: Let $T_v(n)$ be the time complexity for computing the visibility graph G_v. $\mathcal{V}$ is resolved in time $O(T_v(n) + T_{\mathcal{G}_\mathcal{V}}(n)) = O(T_{\mathcal{G}_\mathcal{V}}(n))$. Since we know from Corollary 3.4 that $T_{\mathcal{G}_\mathcal{V}}(n)$ is the lower bound for $\mathcal{V}$, the algorithm is optimal. □

It should be remarked that computation of G_v depends also on the number m of parameters that define σ (see §2.1), while m is not a relevant parameter in the size of $\mathcal{G}_\mathcal{V}$. This means that the theorem can be applied only when m and n are proportional, e.g., when S contains all vertices of the TIN or the grid describing σ, or a constant fraction of them. Theorem 3.6 establishes the complete equivalence between discrete visibility problems and corresponding graph problems, provided that their complexity is high enough. From it, we can derive the following practical rule:

Rule 3.7 *If we are facing a (complex enough) discrete visibity problem, and an efficient algorithm is given for its corresponding graph problem, such an algorithm will efficiently solve also our visibility problem.*

If problems $\mathcal{V}$ and $\mathcal{G}_\mathcal{V}$ are both intractable optimization problems, and a suboptimal polynomial algorithm for $\mathcal{G}_\mathcal{V}$ is known, then such an algorithm can be used successfully for finding a suboptimal solution to $\mathcal{V}$ as well. Since several practical visibility problems prove to be equivalent to intractable graph problems, it is worthwhile to exploit suboptimal solutions of graph algorithms for resolving visibility problems as well. Let us we assume that a criterion of optimality required in both $\mathcal{V}$ and $\mathcal{G}_\mathcal{V}$ consists in minimizing some cost, and that the ratio between the costs of the optimal solution and of any other solution can be measured (see, e.g., §4.2.3 for an example). Hence, we have the following

Rule 3.8 *If $\mathbf{A}_{\mathcal{G}_\mathcal{V}}$ finds a suboptimal solution S_g for $\mathcal{G}_\mathcal{V}$ that is guaranteed to be within some known bound r from the optimal solution of $\mathcal{G}_\mathcal{V}$, then S_g gives a suboptimal solution also for $\mathcal{V}$ that is within the same bound r from the optimal solution of $\mathcal{V}$.*

4 Examples of discrete visibility problems

Following Nagy (Nagy, 1994), we can outline three main classes of visibility problems that have relevant impact in the context of practical applications:

- **Observation points**: problems related to the placement of facilities such as fire towers, radar sites, or scenic sites. In general problems are concerned with either the minimization of facilities needed to see a certain area (in the discrete case, a number of observed objects), or the maximization of the area (number of objects) seen from a single facility.

- **Line-of-sight communication**: problems related to the placement of either antennae or relays to allow sites to communicate by line-of-sight, such as microwave transceivers for telephone, FM radio, television, and digital data networks. In the special case of television broadcasting, relays must be placed in such a way that a signal can be broadcasted from a transmission station to a number of sites on the terrain (e.g., towns): this problem involves aspects of both this class and the previous one.

- **Surface paths**: problems related to the design of paths under visibility constraints, such as either remaining hidden from given observers, or seeing as much landscape as possible. This class includes both static problems - in which a path is designed according to its visibility features with respect to the rest of the terrain, and dynamic problems - in which one or more observers move along paths, being subject to constraints about their visibility.

In the following, we give a non-exhaustive review of relevant problems for each class, and we outline analyses and possible practical solutions that are based on the theory developed in §3. We show that some problems involving optimization are NP-complete. For such problems we outline suboptimal solutions from the theory of graph algorithms.

4.1 Observation points

Problems related to the placement of observation points can be defined in terms of either a finite number of points to be watched, or of an area on the terrain to be observed. In the former case a discrete visibility model defined on $S_v \times S_o$ gives sufficient information for providing and testing a candidate solution to the problem.

4.1.1 Scenic sites

A typical context in which the problem of *scenic sites* (SS) arises is the detection of panoramic points along tourist paths. Such problem can be formulated as the following discrete visibility problem:

> Given a terrain σ, and two sets S_v and S_o of points on σ (respectively called the set of observation points and the set of target points), find the observation point p that sees as many target points as possible.

In terms of visibility graph, SS becomes the following graph problem:

> Given the visibility graph G_v defined on $S = S_v \cup S_o$, find the node $p \in S_v$ with the maximum outgoing degree.

A solution can be found by simply considering each node in S_v and by counting the number of its outgoing arcs. The worst-case time complexity is $O(n_o n_v)$.

4.1.2 Watch towers

The problem of *watch towers* (WT) relies on the surveillance of a certain area by means of a set of towers placed on the terrain: such set is required to be as small as possible, and each point within the area of interest must be seen by at least one tower. Typical application areas include the location of fire towers for forest fire monitoring, or lighthouses for safe navigation. Similar problems arise in military surveillance and weather forecasting. WT can be formulated as the following discrete visibility problem:

> Given a terrain σ and two sets S_v and S_o of points on σ, find a set $S' \subset S_v$ of minimum cardinality, such that any point in S_o can be seen from at least one point in S'.

In terms of visibility graph, we have the following graph problem:

> Given the visibility graph $G_v = (S, E_v)$ defined on $S = S_v \cup S_o$, find a set $S' \subset S_v$ of minimum cardinality, such that for any node $p \in S_o$ there exists at least one node $q \in S'$ auch that $(q, p) \in E_v$.

The continuous version of WT, i.e., to find a minimum number of points that can see the whole surface σ (or a given portion of it), has been considered in (Cole and Sharir, 1989), and its NP-completeness has been shown via reduction to satisfiability (see (Garey et Johnson, 1979)). Here, we show that also the (expectedly simpler) discrete version is NP-complete.

Theorem 4.1 *Problem WT is NP-complete.*

Proof: We consider the following problem on graphs, called the *dominating set* problem (DS): given a graph $G = (N, A)$, find a set $N' \subset N$ of minimum cardinality such that $\forall p \in N - N'$ there exists $p' \in N'$ for which $(p', p) \in A$. DS has been proven to be NP-complete (Garey et Johnson, 1979). DS trivially reduces to WT in polynomial time by imposing $S_o \equiv S_v$, and by noticing that a point always sees itself, thus in the visibility graph $G_v = (S, E)$, $(p, p) \in E$ for each $p \in S$. □

The WT problem, and some of its variations, have been considered in (Lee, 1991): three different heuristics have been proposed. Although no bound is warranted with respect to the optimal solution, the effectiveness of such heuristics is demonstrated by experimental results made on reasonably large data sets.

4.2 Line-of-sight communication

Problems involving line-of-sight communication rely on guaranteeing intervisibility between all pairs of sites of a predefined set. Relays may be permitted or not: whenever permitted, the number of relays should be as small as possible.

In the context of line-of-sight communication, using a discrete visibility model is even more strongly motivated with respect to problems related to the placement of observation points, since the predefined set of points and the set of possible relays are always finite. Furthermore, the choice for relay location is often constrained to strategic points on the terrain.

4.2.1 Communication without relays

In applications in which relays are not permitted, directional antennae are generally used at each site to ensure communication between pairs of sites. The aim is to minimize the total number of antennae. Hence, the line-of-sight communication problem consists of finding a network, whose sites are located in correspondence with a predefined set S', and connecting all points in S'. For each pair (p, q) of points of S', either q is visible from p or there exists a sequence of points connected in the network $v_1, .., v_n$ in S' such that v_1 is visible from p, v_{i+1} is visible from v_i (with $1 \leq i \leq n$), and q is visible from v_n. We call such a sequence a *line-of-sight connected path* between p and q, and the global network a *connected visibility network* for S'. Directional antennae are placed at each site p in S': one antenna for each other site that is connected to p in the network. This yields the formulation of the following discrete visibility problem:

> Given a terrain σ, a set $S = S_v \cup S_o$ of points on σ and a subset $S' \subseteq S$ find a connected visibility network for S' using the minimum number of antennae.

In terms of visibility graph, we have the following graph problem $\mathcal{G}_{\mathcal{V}}$:

> Given the visibility graph G_v defined on a set S and a subset $S' \subseteq S$, find a connected subgraph G' of G_v having S' as its vertex set and an arc set E' of minimum cardinality.

It is immediately seen that $\mathcal{G}_{\mathcal{V}}$ is equivalent to find a spanning tree (in the following we will denote such a problem with ST) on the subgraph G'_v of G_v induced by set S'. ST, and thus $\mathcal{G}_{\mathcal{V}}$, may not have solution: this happens if G'_v is not connected. A polynomial solution can be found as follows: G'_v can be computed in $O(n_o * |S'|)$ worst-case time, since for each node of S', all its outgoing arcs in G_v have to be tested for membership in G'_v (Aho et al., 1983). Connection of G'_v can be tested through a depth-first or a breadth-first visit of G'_v in $O(|E'_v|)$ (where E'_v is the arc setof G'_v). If G'_v is connected, the spanning tree G' for S' can be computed from G'_v, in $O(|E'_v|)$. If G'_v is disconnected, a spanning tree can be determined for each connected component of G'_v in just the same way.

4.2.2 Two-point communication

The *two-point communication* problem (TPC) arises whenever two non-visible points on the terrain are given, and communication has to be warranted between them through relays. Fields of application are all those in which communication between a transmitter and a receiver station is involved (television and radio transmission, electro-magnetic signal exchange, etc.), or in which wayfinding between a departure and a destination site is concerned (navigation or terrain exploration).

Since the number of relays must be finite, a discrete visibility model is sufficient for the formulation of the problem. The following discrete visibility problem $\mathcal{V}$ gives a formalization for TPC:

> Given a terrain σ, a set $S = S_v \cup S_o$ of points on σ and two points $p \in S_v$ and $q \in S_o$, find the minimum subset $S' \subseteq ((S_v \cap S_o) \setminus \{p, q\})$ (called the set of relays) that permits communication between p and q along a line-of-sight connected path.

In terms of visibility graph, we have the following graph problem $\mathcal{G}_{\mathcal{V}}$:

> Given the visibility graph G_v defined on $S = S_v \cup S_o$, and two points $p \in S_v$ and $q \in S_o$, find the minimum subset $S' \subseteq ((S_v \cap S_o) \setminus \{p, q\})$, for which there exists a connected path $(p, v_1, .., v_k, q)$ between p and q with vertices $v_1, .., v_k$, in S'.

Note that if q and p do not belong to the same connected component of G_v, then problem $\mathcal{G}_{\mathcal{V}}$, hence problem $\mathcal{V}$, does not have a solution. As we will see in §4.2.3, $\mathcal{V}$ is a special case of the discrete visibility problem called the line-of-sight network and $\mathcal{G}_{\mathcal{V}}$ is a special case of the corresponding graph problem.

If a solution for problem $\mathcal{G}_{\mathcal{V}}$ exists, it can be solved through a breadth-first visit of G_v starting at node p until point q is reached. Otherwise the visit may stop before q has been met. The time cost is $O(\mid E_v \mid)$, where E_v is the arc set of G_v.

4.2.3 Line-of-sight network

The problem of finding a *line-of-sight network* (LSN) has relevance in several applications, such as the optimal location of radar, laser or sonar surveillance systems, of television transmitters on a predefined area on the terrain, etc. In such contexts a set S^* of transceiver/receiver stations is located on the terrain and auxiliary relays must be placed in such a way that a signal transmitted from any station can be received by any other. Clearly, the number of relays should be as small as possible. This corresponds to determine the minimum set of relays that permit connection of all stations in a connected visibility network. As already mentioned, the two-point communication problem is a special case of the line-of-sight network problem for which set S^* contains only two points.

LSN can be formulated as the following discrete visibility problem:

> Given a terrain σ, a set $S = S_v \cup S_o$ of points on σ and a subset $S^* \subset S$ (called the set of transmitter/receiver stations), find the subset $S' \subseteq ((S_v \cap S_o) \setminus S^*)$ (called the set of relays) of minimum cardinality such that points in $S' \cup S^*$ can be arranged in a connected visibility network.

In terms of visibility graph, we have the following graph problem $\mathcal{G_V}$:

> Given the visibility graph G_v defined on $S = S_v \cup S_o$, and a subset $S^* \subset S$, find the subset $S' \subseteq ((S_v \cap S_o) \setminus S^*)$ of minimum cardinality such that the subgraph of G_v induced by $S' \cup S^*$ is connected.

Problem $\mathcal{G_V}$ is known in graph theory as the uniformly-weighted Minimum Steiner Tree (MST). The undirected version of MST has been proved to be NP-complete (Garey et Johnson, 1979); from Remark 3.1 also the directed version is NP-complete. From Theorem 3.3 we have that MST can be reduced to LSN in polynomial time. Hence, also LSN is NP-complete.

In (De Floriani et al., 1994) a suboptimal graph algorithm was applied for finding an approximate solution to the LSN problem in $O(|S^*||S|^2)$ time in the worst-case, given the visibility graph G_v. Such an LSN algorithm is based on an algorithm for approximate MST proposed by Kou, Markowsky and Berman (Kou et al., 1981). The total cost (i.e., number of relays) of the network computed by such an algorithm is warranted within a factor $2 - \frac{2}{|S^*|}$ of the cost of the optimal solution. Experiments performed on triangulated terrains built from real world data have shown good results, both in terms of computing time and of quality of the approximate solution.

4.2.4 Critical points

Given a set of receiver/transmission stations and a set of relays organized into a connected visibility network, the problem of *critical points* (CP) consists of determining relays that are fundamental for communication, i.e., relays whose failure would interrupt communication. Application fields in which such problem arises are those already mentioned for the LSN, in which failure detection is warranted.

The problem of critical points can be formulated as the following discrete visibility problem:

> Given a terrain σ, a set $S = S_v \cup S_o$ of points on σ, a set $S^* \subset S$ of transmitter/receiver stations and a set $S' \subseteq ((S_v \cap S_o) \setminus S^*)$ of relays such that $S^* \cup S'$ form a connected visibility network, find the points of S' whose failure disconnects the network.

This translates into the following graph problem $\mathcal{G_V}$ on the visibility graph:

> Given the visibility graph G_v defined on $S = S_v \cup S_o$, and two sets S^* and S' as above, such that the restriction G' of G_v to $S^* \cup S'$ is connected, find the nodes of S' that are cut-points for G'.

Note that biconnected components of G' are those maximal subgraphs of G' that remain connected after removing cut-points. Such components correspond to portions of the visibility network whose sites remain in communication after the failure of any critical point.

Cut-points (and associated biconnected components) of a connected graph can be determined in linear time with respect to the number of its arcs (see, for instance, (Gondran et Minoux, 1986)). Thus, solving problem $\mathcal{G}_{\mathcal{V}}$ requires $O(|A'|)$, where A' is the set of arcs of G'.

A generalization of CP is determining k-tuples (with $k \geq 1$)of relays whose failure interrupt communication among receiver/transmission stations in the given network. In terms of visibility graph this corresponds to determine the *k-sets* of G'. Given a connected graph $G = (N, A)$, a k-set $N_c \subseteq N$ is a set of k vertices such that the subgraph derived from G by omitting the vertices of N_c ceases to be connected. Hence, cut-points are 1-sets. 2-sets (and associated 3-connected components) of G' can be determined in $O(|A'|)$ time (Gondran et Minoux, 1986). A *minimum cut-set* for G', i.e., a k-set of minimum cardinality, can be found in $O(|N'||A'|\log|N'|)$, where N' is the set of nodes of G' (Tarjan, 1983).

4.2.5 Fault tolerant network

The *fault tolerant network* problem (FTN) combines the problem of finding a line-of-sight network with the problem of determining critical points. In several applications, such as electrical or telephonic communication, it is important to determine the minimum set of relays that allow joining a given set of transceiver/receiver stations into a connected network with the further constraint of reliability: communication among stations should be warranted even though a site fails. Failure may be due to different operational causes: overload, errors, destruction, etc. In such contexts, two antagonistic goals have to be considered: minimization of total cost and maximization of reliability. Reliability is often expressed in terms of connectivity degree of the network. While the MST connecting the given set of stations represents an optimal solution to the minimization of cost, it can be extremely vulnerable to failures.

FTN can be formulated as the following discrete visibility problem:

> Given a terrain σ, a set $S = S_v \cup S_o$ of points on σ and a subset $S^* \subset S$ (called the set of transmitter/receiver stations), find the subset $S' \subseteq ((S_v \cap S_o) \setminus S^*)$ (called the set of relays) of minimum cardinality such that:
>
> - points in $S' \cup S^*$ can be arranged in a connected visibility network, and
> - the failure of k points in $S' \cup S^*$ (with $1 \leq k \leq (\mid S' \cup S^* \mid -1)$) does not disconnect the network.

FTN becomes the following graph problem $\mathcal{G}_{\mathcal{V}}$ on the visibility graph G_v:

> Given the visibility graph G_v defined on $S = S_v \cup S_o$, and a subset $S^* \subset S$, find the subset $S' \subseteq ((S_v \cap S_o) \setminus S^*)$, of minimum cardinality such that the subgraph of G_v induced by $S' \cup S^*$ is k-connected.

A graph $G = (N, A)$ is *k-connected* (with $1 \leq k \leq\mid N \mid -1$) if for any pair (p, q) of vertices there exist at least k disjoint paths connecting p to q (Winter, 1987). Note that the unique difference with respect to the graph formulation $\mathcal{G}_\mathcal{V}$ for LSN consists of requiring that the subgraph of G_v induced by $S' \cup S^*$ is k-connected instead of connected. For LSN such subgraph is always a tree, while this is not the case for FTN when $k \geq 2$.

This graph problem is known to be NP-complete. According to Winter, no specialized algorithms nor heuristics have been developed, except for the special cases k=2,3 and on graphs with special structural properties (Halin networks, outer planar graphs, etc.) (see (Winter, 1987)).

A similar problem involves considering communication failures, instead of site failures. Failures in communication links are due, for instance, to the placement of obstacles on the terrain that interrupt lines-of-sight. A formulation for such version of FTN on the visibility graph G_v can be obtained from formulation $\mathcal{GV}$, given above, by requiring k-edge-connectivity instead of k-connectivity of the resulting network. The same computational complexity result stated above holds also in this case, and heuristics specific only for special cases have been developed. In particular, an interesting result has been proven for k=2: the optimal traveling salesman cycle through base vertices is not larger than $\frac{4}{3}$ the cost of an optimal solution (see (Winter, 1987) for detailed bibliography on this subject). Thus, solving the traveling salesman problem either exactly or by means of one available heuristics yields a feasible solution to the 2-edge connected version of FTN.

4.2.6 Television broadcast

The *television broadcast* problem (TB) consists of determining the minimum number of relays necessary to connect a given transceiver station to a given set of receiver stations. A minimum number of relays may be used to this purpose. Forthermore, receiver stations cannot be employed as relays: this requirement corresponds to the practical necessity of avoiding having a receiver station located near a relay to ensure good signal reception.

TB can be formulated as the following discrete visibility problem:

> Given a terrain σ, a set $S = S_v \cup S_o$ of points on σ, a point $p \in S_v$ (called the transmitter station), and a subset $S^* \subset S_o \setminus \{p\}$ (called the set of receiver stations), find a subset $S' \subseteq (S_v \cap S_o) \setminus (S^* \cup \{p\})$ (called the set of relays) of minimum cardinality such that for each receiver station q there exists a line-of-sight connected path from p to q.

Since receiver stations cannot be employed as relays, we can consider the visibility graph defined on $S = S_v \cup S_o$, in which we disregard outgoing arcs from nodes in S^*. Thus, G_v is a directed graph whose set of leaves (i.e., of nodes with no outgoing arcs) contains all nodes in S^*. On the visibility graph G_v, TB can be formulated as the following graph problem $\mathcal{G}_\mathcal{V}$:

> Given S, p and S^* as above, find the minimum subset $S' \subseteq (S_v \cap S_o) \setminus (S^* \cup \{p\})$ such that for each $q \in S^*$ there exists a connected path between p and q with vertices in S'.

This problem is equivalent to finding an MST rooted at p on the special directed graph G_v for which the base vertex set S^* coincide with the set of leaves.

We now show that problem $\mathcal{G}_\mathcal{V}$ is equivalent to MST on generic directed graphs.

Theorem 4.2 *$\mathcal{G}_\mathcal{V}$ is equivalent to MST on generic directed graphs.*

Proof: Reduction of $\mathcal{G}_\mathcal{V}$ to MST is immediate, since $\mathcal{G}_\mathcal{V}$ is indeed an MST on the directed graph G_v. Reduction of MST on generic directed graphs to $\mathcal{G}_\mathcal{V}$ can be obtained through the following two steps:

1. Let G be a generic directed graph, let r be a vertex of G (the root for the MST), and let B be a subset its vertices (the base set for the MST). G can be transformed in polynomial time into a directed graph G', having a subset B' of its vertices as leaves, where B and B' are in one-to-one correspondance:
   - G' and B' are initialized, respectively, to G and to B;
   - for each $q \in B$, if q has at least one outgoing edge in G, a new node q' and a new edge e' connecting q to q' are added to G'. Node q is deleted from B' and node q' is inserted into B'.
2. A minimum Steiner tree T' for G' with root vertex r and base set B' is transformed in polynomial time into a minimum Steiner tree T for G with respect to r and with B as base vertex set. Indeed, by definition, T' is rooted at r, it contains a path from r to every node in B', and it has minimum cost. T is obtained by pruning all leaves that are not nodes of B. It is straightforward to see that also T is rooted at r, it contains a path from r to every node in B, and has minimum cost.

□

Since MST on generic directed graphs is NP-complete (see Duin et Volgenant, 1987), also TB is NP-complete. Equivalence allows us to use heuristics proposed in the literature for MST to solve $\mathcal{G}_\mathcal{V}$. In particular, according to Winter, most algorithms and heuristics for the undirected MST can be modified without difficulty to the directed case (Winter, 1987). A heuristic procedure for obtaining feasible solutions have been proposed by Wong (see (Wong, 1984)): experimental results have been reported that show the effectiveness of the method on reasonably large graphs.

4.3 Surface paths

Finding surface paths implies the determination of lines on the terrain with specific visibility characteristics (such as scenic peculiarities, invisibility, etc). For this reasons, the solution of such problems involves retrieval of both visibility and adjacency information about points on the terrain. Since a line on the terrain is formed by an infinity of points, continuous visibility models seem more suitable in this context. However, discrete approximations can be used as well. A discrete visibility model is appropriate if, for instance, a discrete gridded model is used for terrain represention and lines on the terrain are approximated by sequences of either 4- or 8-connected paths on the gridded model. A discrete visibility model can also be used whenever the topographic surface is represented by a TIN and lines are approximated by sequences of edges of the TIN. In both cases, reducing such problems to graph problems involves both the visibility graph and the *adjacency graph* encoding adjacency relations among points on terrain. In particular, for a gridded model, pixel-to-pixel adjacencies should be stored, while for a TIN vertex-to-vertex adjacencies should be encoded. In the following, we will denote the adjacency graph with G_t, and we will assume that the set S of nodes of G_v is always a superset of the set of nodes of G_t. Since all pairs of adjacent points are mutually visible, a fortiori G_t is a subgraph G_v.

4.3.1 Hidden path

The problem of *hidden path* (HP) consists of determining a path on the terrain from a given site a to a given site b that is as hidden as possible from a given set of observation points. Such problem finds applications in several contexts, such as scenic landscape assessment, terrain exploration, and military topography.

Let us refer to a path on the terrain as a *topological path* to distinguish it from a line-of-sight path. HP can be formulated as the following discrete visibility problem:

> Given a set S_o of points on a terrain σ, two points a and b in S_o, and a set of points S_v on σ (the observation points), find a connected topological path between a and b having vertices in S_o such that the number of vertices of the path visible from S_v is minimum.

HP can be formulated on the visibility graph G_v and on the adjacency graph G_t as the following graph problem $\mathcal{G_V}$:

> Let S_o, a, b and S_v as before. Let G_v be the visibility graph defined on $S_o \cup S_v$ and G_t be the adjacency graph, defined on S_o. Find a connected path on G_t between a and b having vertices in a subset S' of S_o such that the number of nodes $p \in S'$ for which there exists an arc (p', p), with $p' \in S_v$, is minimum.

A polynomial time solution to problem $\mathcal{G_V}$ is based on the following two steps:

1. Graph G_v is visited: nodes in S_o that are visible from at least one node in S_v are assigned a constant positive weight, while non-visible nodes are assigned a null weight.

2. A minimum cost path between a and b is determined on the adjacency graph G_t. The function cost to minimize is the sum of weights of nodes included in the path. This step can be accomplished through the well-known Dijkstra's algorithm. In fact, Dijkstra's algorithm works on directed graphs with weights on arcs. Given a graph G with weighted nodes, it can be transformed into a graph G' with weighted arcs by simply assigning to every arc $e = (v_1, v_2)$ the sum of the weights of v_1 and v_2. Dijkstra's algorithm on G' produces the required solution for G.

Time complexity of step 1 is $O(n_v|S_o|)$, since for each visited node of graph G_v belonging to S_o, visibility from nodes of S_v has to be tested. Step 2 requires $O(|S_o|^2)$ time (see (Aho et al., 1983)).

4.3.2 Scenic path

The problem of *scenic paths* (SP) involves the determination of a path between two given sites that is particularly interesting from a panoramic point of view. This problem can be formulated as an optimization problem $\mathcal{G}_{\mathcal{V}}$, in which the number of "boring" points, from which no scenic sites are seen, is minimized.

> Let S_v be a set of points on a terrain σ; let a, b be two points of S_v and S_o be a set of points on σ. Let G_v be the visibility graph defined on $S_v \cup S_o$ and G_t be the adjacency graph defined on S. Find a connected path on G_t between a and b having vertices in a subset S' of S, and such that the number of "boring" nodes $p' \in S_o$ for which there exists no arc (p, p'), with $p \in S'$, is minimum.

The graph problem is completely analogous to the one in the previous case, on graph G_t where a weight 0 is assigned to all nodes that see at least one scenic site, while a weight 1 is assigned to all remaining points. A more refined formulation could assign a weight inversely proportional to the number of scenic sites seen from any viewpoint.

A converse formulation of the problem, i.e., to maximize the number of scenic sites, is not well-defined unless reasonable further requirements on the lenght of the topological path between a and b are added. In this case, the problem is likely to become considerably more difficult, since optimization must be performed with respect to contrasting requirements. We have no knowledge about the complexity of the problem, nor about possible heuristics for approximated solutions.

5 Practical issues

Finding an appropriate set of representative points to build a discrete visibility model for practical purposes is a non-trivial problem. Sets too large could make algorithms not useful in practice, while sets too small would produce results that have no practical meaning. Though several studies have been made in order to extract terrain features, which could be used as representative points, it is still not clear what is a "good" choice. In the experiments performed in (De Floriani et al., 1994) and (Lee, 1991), the whole set of vertices of a TIN have been used as representative points. This is a reasonable choice for line-of-sight communication and observation point problems, as long as terrain is represented by a TIN of reduced size (say, a few thousands vertices), while it can be impractical for large TINs. Similarly, it is often impractical to use all pixels of a gridded model as representative points.

On the other hand, some problems, like the ones involving computation of paths, would require a continuous representation and analysis of terrain, thus involving in principle an either continuous, or at least semi-continuous visibility model. As we mentioned in §2.1, also discrete approximations of such models, based on gridded elevation models, require using the whole set of pixels as either viewpoints or observed points.

Major drawbacks in handling large representative sets are in computing and storing the visibility graph G_v. Moreover, complete computation of G_v often represents the bottleneck in discrete visibility algorithms. A possible remedy to such drawbacks is using a dynamic approach in computing G_v. Such an approach was proposed for TPC and LSN in (De Floriani et al., 1994): G_v is considered virtually available, while the visibility from a given point is actually computed and stored only when (and if) such information is needed. Although such a solution in general does not improve the worst-case time behavior and space requirement of an algorithm for a specific visibility problem, the improvement in practical performance can be dramatic. However, when several instances of visibility problems must be resolved on the same terrain and on the same set of representative points, it might be convenient to compute G_v once and for all and to store it entirely. Hence, we have a last rule of thumb for designing discrete visibility algorithms:

Rule 5.1 *If several instances of a visibility problem must be resolved on the same dataset, and the storage of the complete visibility graph is feasible, then compute G_v first, and use it for resolving all instances. If only one (or few) instance(s) of the problem must be resolved, and/or the set of representative points is too large to permit to store G_v, then use a dynamic approach.*

6 Concluding remarks

We have given some theoretical results that show how discrete visibility problems, especially optimization problems, can be interpreted and studied as graph problems. We have put such results into practical rules that can be followed to find graph solutions to visibility problems. Moreover, we have translated a number of visibility problems of practical interest into graph problems known in the literature, and we have outlined some graph algorithms for their solution, either exact or approximate.

Our results suggest that an exploration of the existing literature on graph algorithms could provide efficient solutions to complex visibility problems. The following books are useful references for this purpose: (Reingold et al., 1977, Even, 1979, Gondran and Minoux, 1986, van Leeuwen, 1990). Also, results about the NP-completeness (intractability) of some visibility problems, found through their equivalence to graph problems, give theoretical support to the use of heuristics or suboptimal algorithms for practical solutions. The best reference for the study of NP-completeness is (Garey and Johnson, 1979).

Further visibility problems might be studied in the context of our framework. For instance, an important class of problems of practical impact is the class of dynamic path problems. Such problems are formulated under the assumption that observers and targets are all moving on the terrain. Different dynamic path problems have been considered in (Teng et al., 1992, Teng, 1993), and heuristics have been proposed, which use parallel computation in the framework of a discrete visibility model built on a gridded elevation model. To our knowledge, no equivalence of problems in this class to graph problems has been investigated so far.

Our graph approach does not help resolving all visibility algorithms, not even all algorithms that involve a finite number of viewpoints and observed points. Examples of visibility problems that cannot be faced in our framework are: problems in which the height of towers at viewpoints or observed points are variables (Lee, 1991, Sharir, 1988); problems in which the visibility of points is not limited to Boolean values (visible/non-visible), but can admit additional information, including, e.g., elevation above or below horizons, uncertainty, etc. (Fisher, 1994); and problems in which also the distance between mutually visible points is relevant.

It would be interesting to extend our approach to some such problems by using weighted visibility graphs, where the cost of each arc corresponds, e.g., to a rate of visibility, or to the length of the line-of-sight. However, our theoretical results cannot be generalized immediately to these cases, because it is not clear how to transform a generic weighted graph into a weighted visibility graph.

Acknowledgement

Support of the National Research Council of Italy under project "Models and Systems for Environmental and Territorial Data Processing" is gratefully acknowledged. We wish to thank Leila De Floriani for initiating the research on LSN, and for suggesting its equivalence to MST, as well as for many useful discussions on the subject of this paper.

References

P.K. Agarwal, M. Sharir, (1994) On the number of views of polyhedral terrains, *Discrete and Computational Geometry*, **12**, pp.177–182.

A.V. Aho, J.E. Hopcroft, J.D. Ullman, (1983) *Data structures and algorithms*, (Reading, MA, USA: Addison-Wesley) pp.203–208.

R. Cole, M. Sharir, (1989) Visibility problems for polyhedral terrains, *Journal of Symbolic Computing*, **7**, pp.11–30.

T.H. Cormen, C.E. Leiserson, R.L. Rivest, (1990) *Introduction to Algorithms*, (Cambridge, MA, USA: MIT Press).

L. De Floriani, P. Magillo, (1994) Visibility algorithms on triangulated terrain models, *International Journal of Geographical Information Systems*, Vol.8, No.1, pp.13–42.

L. De Floriani, P. Marzano, E. Puppo, (1994) Line-of-sight communication on terrain models, *International Journal of Geographical Information Systems*, Vol.8, No.4, pp.329–342.

C., Duin, W., Volgenant, (1987) Some Generalizations of the Steiner Problem in Graphs, *Networks*, **17**, pp.353-364.

S. Even, (1979) *Graph Algorithms*, (Potomac, MD: Computer Science Press).

P. Fisher, (1994) Stretching the viewshed, in T.C. Waugh, R.G. Healey, Eds., *Proceedings 6th International Symposium on Spatial Data Handling*, Edinburgh (UK), September 5-9, 1994, (Columbia, SC, USA: IGU Commission on GIS – London, UK: Association for Geographic Information), pp.725–738.

M.R. Garey, D.S. Johnson, (1979) *Computers and Intractability - A guide to the theory of NP-completeness*, (San Francisco, CA, U.S.A.: Freeman et Co.).

Z. Gigus, J. Canny, R. Seidel, (1991) Efficiently computing and representing aspect graphs of polyhedral objects, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **13**(6), pp.542-551.

M., Gondran, M., Minoux, (1986) Graphs and Algorithms, (Wiley-Interscience Series in Discrete Mathematics: John Wiley et Sons).

L. Kou, G. Markowsky, L. Bermann, (1981) A Fast Algorithm for Steiner Tree, *Acta Informatica*, **15**, pp.141–149.

J. Lee, (1991) Analyses of visibility sites on topographic surfaces, *International Journal of Geographical Information Systems*, **5(4)**, pp.413–429.

G. Nagy, Terrain visibility, 1994, *Computers et Graphics*, **18**(6), (Oxford, UK: Pergamon), pp.763-773.

F.P. Preparata, M.I. Shamos, (1985) *Computational geometry: an introduction*, (New York, NY, U.S.A.: Springer-Verlag).

E.M. Reingold, J. Nievergelt, N. Deo, (1977) *Combinatorial Algorithms: Theory and Practice*, (Englewood Cliffs, NJ: Prentice Hall).

M. Sharir, (1988) The shortest watchtower and related problems for polyhedral terrains, *Information Processing Letters*, **29**, (Elsevier), pp.265-270.

R.E. Tarjan, (1983) *Data Structures and Network Algorithms*, (Society for Industrial and Applied Mathematics, Philadelphia, PA, U.S.A.).

Y.A. Teng, D. DeMenthon, L.S. Davis, Stealth terrain navigation for multi-vehicle path planning, (1992), *International Journal of Pattern Recognition and Artificial Intelligence*, 6, pp.395-415.

Y.A. Teng, Parallel processing of geometric structures: visibility and triangulation algorithms, (1993), *PhD Thesis*, (College Park, MD, U.S.A.: Department of Computer Science, University of Maryland), also published as *Technical Report* CAR-TR-680, (College Park, MD, U.S.A.: Center for Automation Research, University of Maryland).

J. van Leeuwen, (1990) Graph algorithms, in *Handbook of Theoretical Computer Science - Volume A: Algorithms and Complexity*, J. van Leeuwen (Editor), (Amsterdam: Elsevier) pp.525-631.

Winter, P., (1987) Steiner Problems in Network: A Survey, *Networks*, **17**, pp.129-167.

Wong, R.T., (1984) A dual ascent approach for Steiner tree problems on a directed graph, *Math. Program.*, **28**, pp.271-287.