Untangling the Prometheus Nightmare

C. Bose* R. Bryce†and G. Dueck‡

April 3, 2009

Abstract

A number of bush and forest fire simulators have been developed in the last two decades based on the deterministic spread equations of Gwynfor Richards (Int. J. Num. Meth. Eng. 1990) and the marker method for computing discrete approximations to the Richards equations. Unfortunately, all such implementations are plagued by topological complications. For example, sampling issues on the evolving front, represented in the Canadian simulator Prometheus as a set of polygons, lead to tangling and other non-physical singularities.

Recently, an automated untangling routine has been developed for Prometheus based on the so-called 2-colour Theorem. Not only is this approach more intuitive than previous algorithms (which were based on scan lines and winding number calculations), it has proved to be more accurate and faster on all test cases employed by Prometheus developers (from 10 to 90 percent speedup compared to previous generation codes, depending on the examples). It is based on a concise and easy-to-implement set of rules which do not introduce the many 'special cases' that previous methods required.

This report presents a brief review of fire simulation models in general, and on their various approaches to untangling in particular. We describe the mathematical foundations for our new method and give a brief overview of the programming issues that arise in implementation. Finally, we report the results obtained by the new model on a test database of fires used by Prometheus developers. Performance is assessed both in terms of run times and in the ability of the code to produce realistic fire fronts without operator intervention.

Keywords: Forest fires, Bushfires, Spread Models, Marker Method, Untangling, Polygon Clipping

*Department of Mathematics and Statistics, University of Victoria, CANADA
†Heartland Software Solutions Incorporated, CANADA
‡Department of Mathematics and Computer Science, Brandon University, CANADA

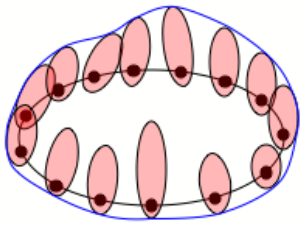

Figure 1: Fire front Evolution via Huygens' Principle

1 Introduction and Presentation of Results

Prometheus, the Canadian wildland fire growth simulator [14] is a a fully-functional, field tested computer program based on a set of continuous spread equations derived from Huygens' Principle. Huygens' Principle proposes that each point on the active fire front serves as the source for a small *firelet*. These firelets are permitted to evolve independently for a short period of time after which the resulting envelope determines the evolved fire front.

This model lends itself to simple numerical simulation where, roughly speaking, the envelope of a large (but finite) number of firelets at time t, evolved over a small positive time Δt forms the fire front at the next time step $t + \Delta t$. Figure 1 shows a pictoral representation of the model.

The Canadian Fire Behaviour Prediction (FBP) system [3], a basic input into the program, assumes an ellipsoidal shape for wildland fires. Richards' differential equations [9, 11] implement Huygens' Principle as discussed above and transform elliptical fire shapes defined by the FBP standard into complex fire shapes modelled over heterogeneous spatial conditions. The data (i.e. fuel, weather, topography) at each vertex serve to dimension and orient an elliptical firelet for the purpose of fire front propagation. The perimeters of the individual firelet are determined over the given positive time step and combined in a way so as to produce a discrete approximation to the Richards' equations (see [9] for details). In the PDE literature this approach is variously known as as the *marker method* or *bead method*[1] and is well-known to suffer various topological complications. The essential problem is that variation in the local velocity field combined with spatial and temporal discretisation leads to tangles and singularities in the computed front. In order to use this method in an interative scheme, it is essential to identify and remove these artifacts by post-processing the computed front before the next iteration. In the literature, this is known as *delooping* or *untangling*; we adopt the latter terminology. In Prometheus, this step forms part of the *vertex management* module.

All versions of Prometheus have incorporated some level of automated untangling. Prior to version 5.0 the algorithm for untangling was based on a winding number[2] calculation performed via a scan line method as proposed in [10]. Once the winding numbers of vertices[3] are determined, various heuristics are applied to assign each vertex either an *active* or *inactive* status. The front is simplified by removing strings of inactive vertices and associated edges. Unfortunately, artifacts in the form of stubborn loops and crossings and stranded interior polygonal segments typically remain after the version 4.5 untangling as shown in the left panel of Figure 2.

Prometheus 5.0 introduced a new module for vertex management containing an untangling routine based on the 2-colour Theorem. This approach is both simpler and faster than previously implemented algorithms and gives more realistic results. Applying the new algorithm to the data generating Figure 2 yields the untangled plots in the right-hand panel. Note that these fronts are free of topological complications and are in reasonable large-scale agreement with the results computed by version 4.5.

One early success with version 5.0 was correct resolution of a stubborn tangle known as the Prometheus Nightmare. This test case had inexplicably resisted solution by all routines prior to

[1] See, for example, J. Sethian, [13] for a discussion and history of this problem

[2] An elementary construction using contour integrals in the theory of complex variables. See [7], for example

[3] Strictly speaking, winding numbers cannot be computed at points on the front – the implementation uses an extended definition of winding number to handle vertices.

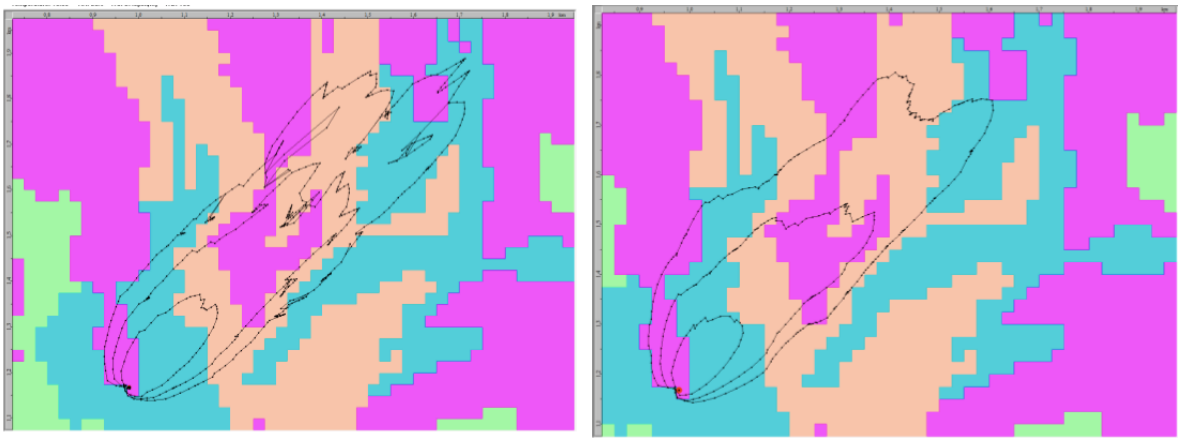

Figure 2: Front evolution via Prometheus 4.5 (left) and Prometheus 5.3 (right)

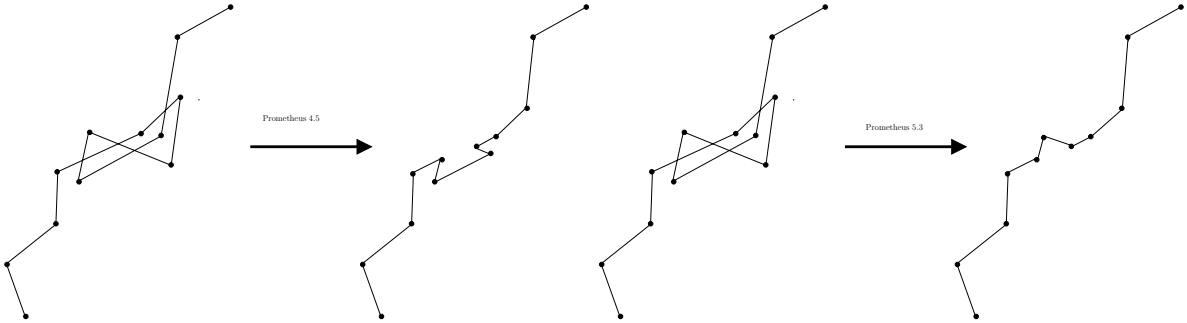

Figure 3: The nightmare untangled under (left) version 4.5 and (right) version 5.3

version 5.0. In Figure 3 the results of the two routines are shown on the nightmare tangle (left: incorrect – Prometheus 4.5 and right: correct – Prometheus 5.3).

2 Timing

The new algorithm is faster than previous vertex management schemes based on winding number. We present some comparative speed results on a set of test problems.

Test Dataset	Version 4.5.2	Version 5.3.0	Percentage Speedup
Grande Cache	765	80	89%
Dogrib	40	35	13%
Mount Somers	50	40	20%
Chisholm	45	20	56%
Quebec Lakes	375	140	63%
Jasper	1950	210	89%

Run times (in seconds) for standardized test datasets

Remark 1.

- Startup overhead dominates in short simulations. This can reduce the percentage speed gains in small fires.
- Field users report increased speed of approximately 30% in version 5.0 over previous releases. This is despite of higher overhead from more elaborate routines written into other parts of version 5.0.
- The timings reported above were performed using a computer with a P8400 processor running @ 2.26 GHz with 4GB RAM.

3 Brief Overview of the Model

3.1 Forest Fire Modelling

Numerous wildfire simulators have been developed via Huygens' Principle and the marker method for front evolution. An early example was presented by Sanderlin and Sunderson [12] . Some implementations assume non-elliptic basic fire shapes such as tear-drops, double ellipses or lemniscates [15]. M. Finney's FARSITE [1] represented a landmark achievement for elliptical fires and remains operational to this day. FARSITE incorporates a variety of interesting heuristics and algorithms to improve speed and accuracy. Finney's approach also addresses tangles, but is computationally very expensive and cannot be parallelized.

To address performance concerns, Finney developed a model based on minimum travel times [2]. This method eliminates tangles, and is highly parallelisable. Because this model, however, solves for time, temporal variations (e.g. changing weather conditions or stochastic spotting) are currently not supported. Migrating Prometheus to this MTT approach, therefore, does not seem feasible.

Cellular automata models are natural and have proven useful in some situations, but generally produce artefacts such as 'coordinate ghosting' that are undesirable [10]. Level set methods [13] are a mathematically attractive approach and an active area of research due to their numerical efficiency and lack of topological complications.

3.2 A brief review of the untangling literature

It is worthwhile to compare the untangling problem presented above with existing solutions in the literature. Generally, these are applied to the problem of *polygon clipping* where one (simple, closed) polygon (the clip) is used to remove an area from another polygon (the subject). This setting, while natural and extremely important in computer graphics, is too restrictive to handle the complex polygons arising from front propagation. Examples we have considered are Vatti's algorithm [16] and its more modern extended heuristics (in [8]) and Weilers algorithm [17], which assumes a 'clockwise' orientation on each polygon. We have compared our problem with the Greiner-Horman algorithm [4] and a recent extension of their method due to Kim and Kim [6]. Of the three, this is closest in spirit to our 2-colour solution but really only applies to the clip and subject setting. None of these methods appear to be easily implemented for our problem within their design parameters.

4 Definitions and Assumptions

4.1 Polygons and crossings

The fire front is assumed to comprise one or more polygons, each determined by an ordered list of vertices $P_i = (x_i, y_i)$, $i = 1, 2, \ldots N$ where $P_1 = P_N$, closing the polygon. We also allow repeated vertices: $P_i = P_j$ for some $1 < i < j < N$. The edges of each polygon are directed line segments in the plane connecting P_{i-1} to P_i. To keep things simple, we will consider only the case of a single polygon in this article.

We assume that at timestep $t = T$ the polygon is untangled, and oriented so that the burned area lies to the left, given the orientation in the usual sense of positive orientation of simple closed curves in the plane. We call this a simple polygon. As discussed above, the propagation algorithm acts on each vertex $P_i(T)$, moving it to $P_i(T + \Delta t)$ according to the (discrete) Richards' equations. This presents us with a new oriented polygon with edges connecting $P_{i-1}(T + \Delta t)$ to $P_i(T + \Delta t)$ which we generally expect to be tangled, i.e. non-simple. Our goal is to replace this with a finite union of simple, disjoint polygons such that the active fire front is accurately represented by the union of the remaining edges and with the burned area laying to the left.

A *crossing* in the polygon is a point of intersection between two or more edges. This includes the possibility of a vertex meeting an edge, or two vertices occupying a common point.

4.2 Resolving Crossings

The algorithm is based on the following simple observation. Every crossing in the polygon creates an ambiguity in the assignment of the status 'burned' or 'unburned' to regions near the crossing. Figure 4 makes the problem clear, where the crossing of two oriented segments is shown on the left. Burned areas are indicated by black shading and unburned (or currently burning) by red shading. The crossing determines (locally) four disconnected components. While two of the components are consistently labelled as burned or unburned, the other two necessarily have ambiguous labels.

We remove the crossing by creating two new vertices at the crossing point, labelled R_1 and R_2 and paths as shown in the left panel of Figure 4. In the figure we have labelled regions near the crossing with 0's and 1's in order to distinguish two methods of resolution. Edges are re-oriented so as to resolve the ambiguity in burned/unburned regions and leave the region labeled 1 to the left of the oriented curves .

Such a labeling the connected components forming the complement of the polygon with 0's and 1's forms an essential part of our algorithm. We are assisted in this by the following elementary result.

Theorem 1. *(2-Colour Theorem [5]) Every piecewise continuous, closed planar path admits a 2-colouring*[4] [5]*.*

We choose a 2-colouring such that the infinite connected component is assigned the label 0. Then we have

Rule 1. Uncrossings are chosen so as to connect regions having label 1. That is, make the lower choice in Figure 4 left pane.

Each uncrossing induces a reordering of a portion of the polygon. In the right-hand panel of Figure 4 we show schematically how the two possible reorderings arise. Note that it is possible that after application of the crossing rule, we are left with two polygons. These can recombine to a single polygon later through application of the crossing rule, or not. In any event, after each application the total number of crossings (counting both self-crossings and crossings between distinct polygons) is reduced.

Theorem 2. *Finitely many applications of the Rule 1 will result in a finite number of disjoint (non-crossing) oriented simple polygons which divide the plane into a single bounded component with label 1 and a finite number of connected components labeled 0. Polygons are positively oriented with respect to the 0-1 labeling.*

Proof. (Sketch). Obtain the 2-colouring as in Theorem 1, arranging the colouring so that the infinite component is labeled 0. Rule 1 ensures that after untangling there will be a single bounded and connected component with label 1. Locally, edges been re-oriented to satisfy the local rule that the label 1 lies to the left of the curve. The 2-colouring will ensure that this rule is satisfied globally. Since there are no crossings, each component is bounded by a simple polygon. □

[4] A 2-colouring is an assignment of labels (which we may as well assume are 0 and 1) to connected components of the complement of the curve in such way that adjacent components sharing a common segment of the curve get distinct labels. Clearly such a colouring is unique up to a flip $0 \leftrightarrow 1$

[5] If the polygon is simple, this is the celebrated Jordan Curve Theorem

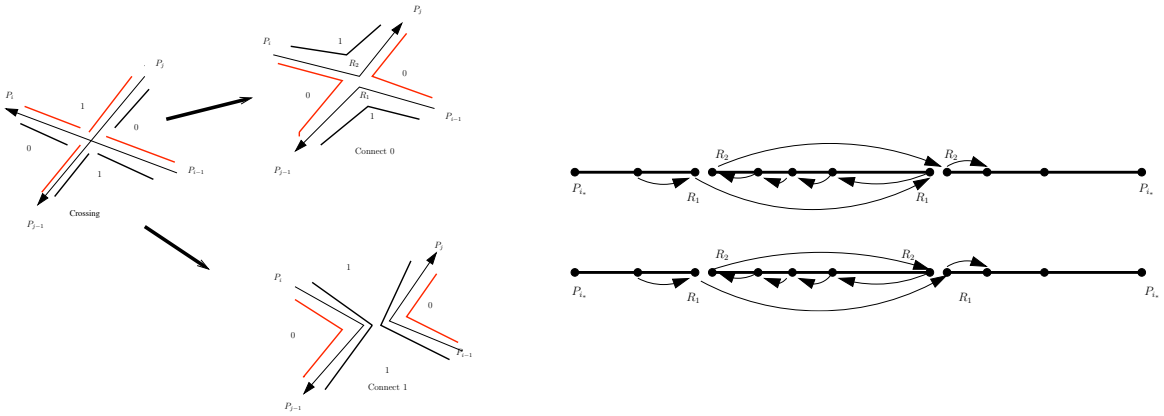

Figure 4: Resolution of crossing point. Geometric resolution (left) and data structure (right).

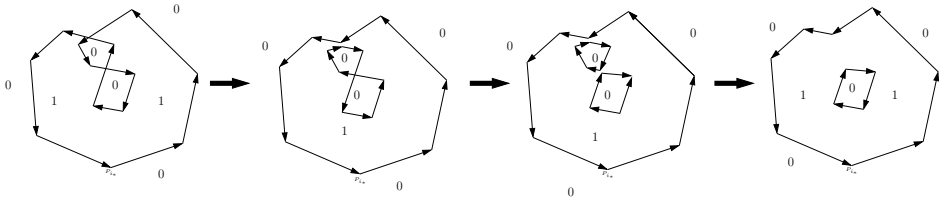

Figure 5: Horseshoe fire

4.3 Clipping the untangled polygons

Clipping is performed on the output from Theorem 2. First, the outer polygon is identified. This is the polygon adjacent to the infinite connected component and contains the extreme bottom-most vertex from the original polygon (easily and reliably identified).

Rule 2. The outer polygon is retained.

The inner simple polygons are retained or clipped according to

Rule 3. Inner polygons are removed if the direction of any edge in that polygon has been reversed when compared to its direction in the original polygon.

Figure 5 shows a simple application of the algorithm in a simple horseshoe fire. The actual implementation untangles one vertex at a time, starting from the vertex P_*, the bottom-most vertex, and working around the polygon in the natural orientation. The untangling leaves two active fire fronts, the outer expanding front and an inward burning 'island'. This is the correct untangling of the horseshoe fire.

5 Conclusions

The algorithm is a significant improvement over previous version untanglers. It is based on a concise and easy-to-implement set of rules which do not introduce the many 'special cases' that previous methods required. However, it is not infallible. For example, it has been noticed that it fails to give correct results on a simple triple-loop example. The difficulty appears to be in the clipping heuristic

contained in Rules 2. and 3. and not in the untangling part of the routine. It is possible that a modified heuristic using the more detailed winding number (the 2-colouring can be determined by winding number, but throws away much of the information) could address examples such as this.

5.1 Acknowledgements

The authors are pleased to acknowledge support from the Alberta Sustainable Resources Department, MITACS, PIMS and the GEOIDE NCE. It is a pleasure to thank C. Tymstra for his support and helpful comments during the course of this work.

References

[1] Finney, M. A., FARSITE: fire area simulator model development and evaluation. USDA For. Serv. Res. Pap. RMRS-RP-4. 1998.

[2] Finney, M. A., Fire growth using minimum travel time methods. Can. J. For. Res. 32: 1420-1424. 2002.

[3] Forestry Canada Fire Danger Group, Development and structure of the Canadian Forest Fire Behaviour Prediction System. Inf. Rep. ST-X-3. For. Can., Ottawa, Ontario, 1992.

[4] Greiner, G., Hormann, K., Efficient Clipping of Arbitrary Polygons. ACM Trans. On Graphics, 17(2) 71-83 1998.

[5] Hunter, D., The two-colour theorem.
`http://blue.butler.edu/~phenders/sigcse2004niftyworkshopMAA_PREP_Workshop_2004/Two\%20Color%20Theorem/twocolor.pdf`

[6] Kim and Kim, An extension of polygon clipping to resolve degenerate cases. Comp. Aided Design and Appl., v3, 447-456. 2006.

[7] Krantz, S. G., The Index or Winding Number of a Curve about a Point. §4.4.4 in Handbook of Complex Variables. Boston, MA: Birkhäuser, 1999, 49-50 .

[8] Murta, A., A General Polygon Clipping Library.
`http://www.cs.man.ac.uk/~toby/alan/software/gpc.html, 1999.`

[9] Richards, G. D., An elliptical growth model of forest fire fronts and its numerical solution, Int. J. Num. Methods Eng. 30: 1163:1179, 1990.

[10] Richards, G. D. and Bryce, R. W., A computer algorithm for simulating the spread of wildland fire perimeters for heterogeneous fuel and meteorological conditions, Int. J. Wildland Fire 5(2) 73-79, 1995.

[11] Richards, G. D., The mathematical modeling and computer simulation of wildland fire perimeters for heterogeneous fuel and meteorological conditions, Int. J. Wildland Fire, 9(3): 213-221, 1999.

[12] Sanderlin, J.C. and Sunderson, J.M., A simulation for wildland fire management planning support (FIREMAN). Vol. 2. Prototype models for FIREMAN (Part II): Campaign fire evaluation. Mission Research Corp. Contract 21-343, Spec. 222. 249 p. 1975.

[13] Sethian, J. Level Set Methods and Fast Marching Methods. Cambridge University Press. 1999.

[14] Tymstra, C., Bryce, R.W., Wotton, B.M. and Armitage, O.B., Development and structure of Prometheus: the Canadian wildland fire growth simulation Model. Nat. Resour. Can., Can. For. Serv., North. For. Cent., Edmonton, AB. Inf. Rep. NOR-X-417. To appear 2009.

[15] Van Wagner, C. E., A simple fire growth model. For. Chron. v45, pp103-104, 1969.

[16] Vatti, B. R. A generic solution to polygon clipping, Commun. ACM 35, 56-63. 1992.

[17] Weiler, K. Polygon comparison using a graph representation, SIGGRAPH 80, 10-18, 1980.