MSc projects at BT Research

Keith Briggs Keith.Briggs@bt.com

keithbriggs.info

University of Bath MSc talk 2007 Dec 12 1400

typeset 2007 December 6 13:25 in pdfLaTeX on a linux system

Adastral Park, Martlesham, Suffolk

- Cambridge-Ipswich high-tech corridor
- 2000 technologists
- 15 companies
- UCL, Univ of Essex

BT Research centres

- Broadband Centre
- Foresight Centre
- IT futures research Centre
- Intelligent systems Centre
- Mobility Centre
- Networks Centre
- Pervasive ICT Centre
- Security Centre
- Asian Research Centre
- Disruptive lab at MIT

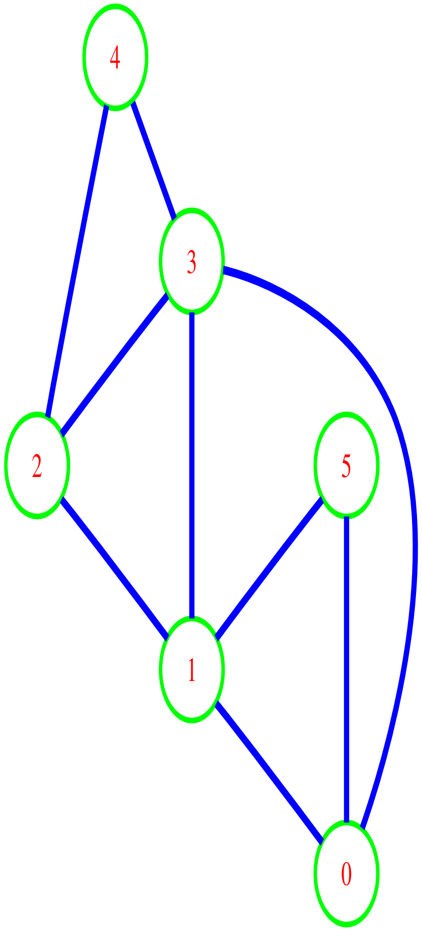
MSc projects supervised by Keith Briggs

Details: `www.keithbriggs.info/MSc_project_ideas_2008.html`

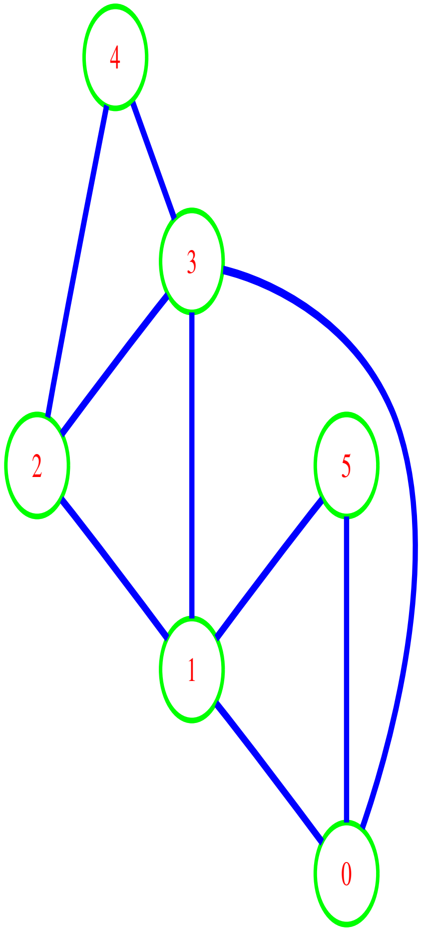
Mostly in discrete maths, two in ODEs. . .

- *1. Semidefinite programming and graph theory*
- *2. Singular value decomposition updating*
- *3. Singular value decomposition methods for Lyapunov exponents*
- *4. Automatic differentiation (AD) methods for ODE sensitivity analysis*
- *5. Fast random selection*
- *6. Random sampling of set partitions*
- *7. Fast counting*
- *8. Random sampling of unlabelled structures*
- *9. Convex optimization in python*
- *10. Geographical computations*
- *11. Statistics of Roman roads in Britain*

Graph concepts

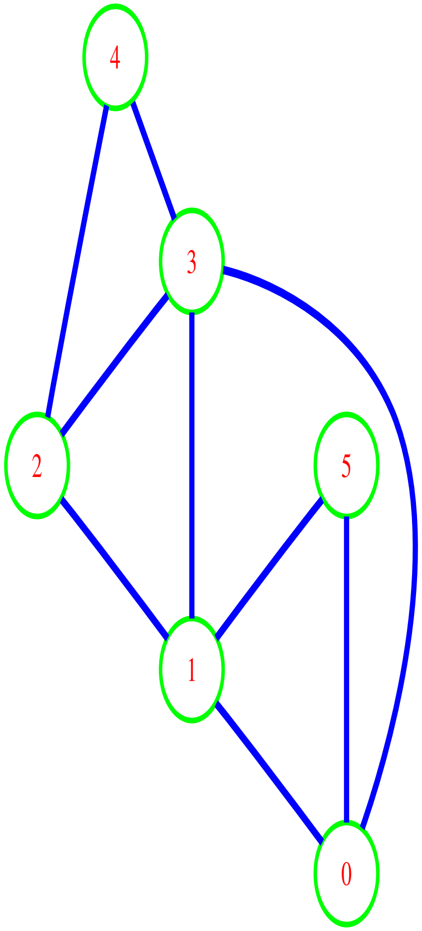


Graph concepts



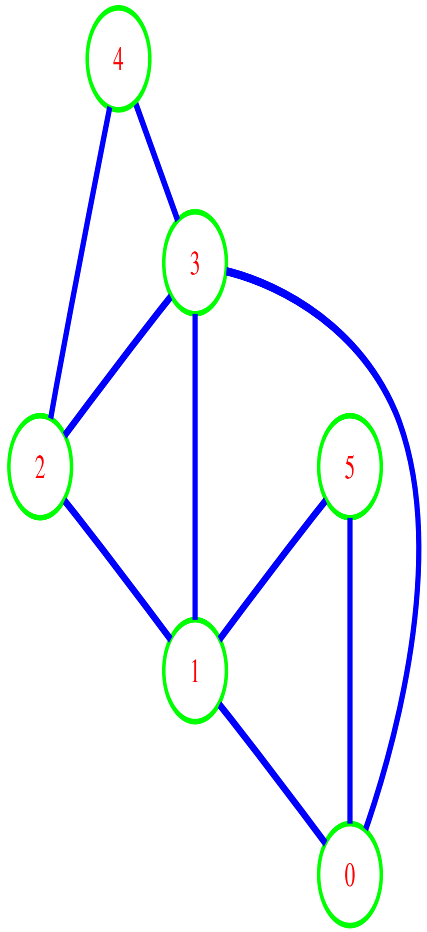
- *clique* - a complete subgraph

Graph concepts



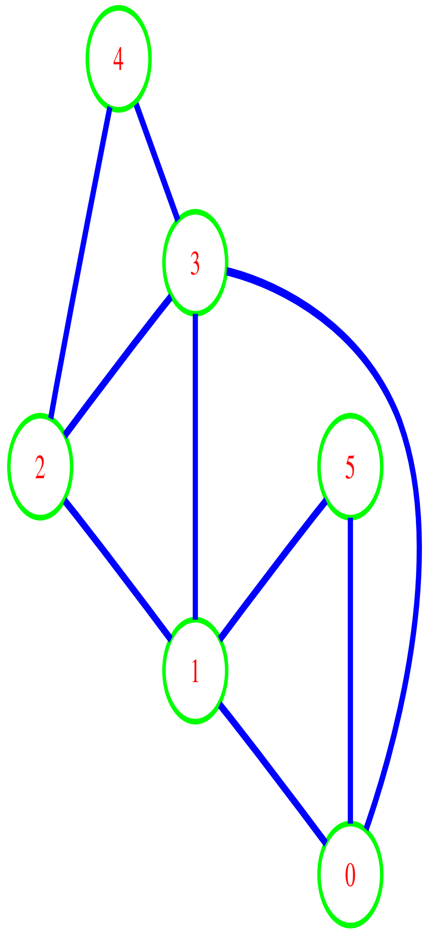
- *clique* - a complete subgraph
- *clique number* ω - the number of nodes in a largest clique

Graph concepts



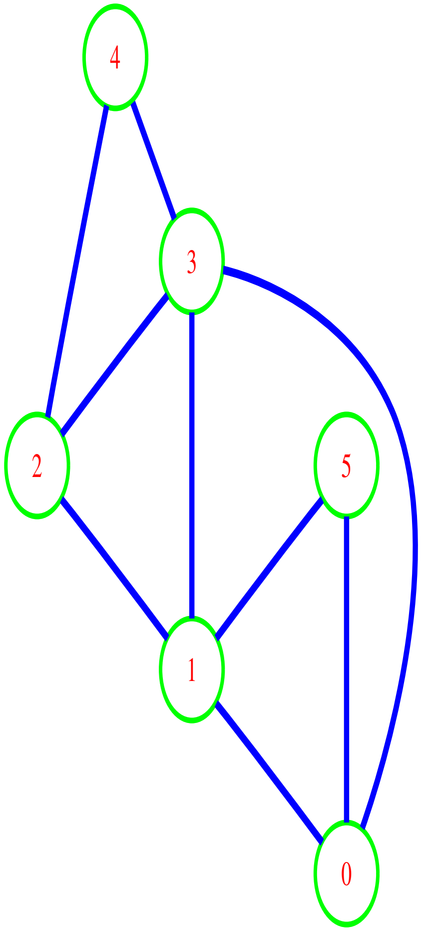
- *clique* - a complete subgraph
- *clique number* ω - the number of nodes in a largest clique
- *colouring* - an assignment of colours to nodes in which no neighbours have the same colour

Graph concepts



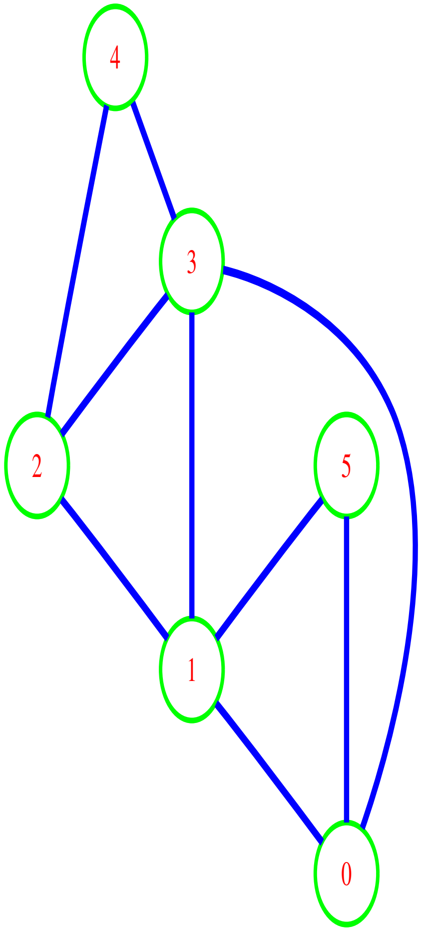
- *clique* - a complete subgraph
- *clique number* ω - the number of nodes in a largest clique
- *colouring* - an assignment of colours to nodes in which no neighbours have the same colour
- *chromatic number* χ - the number of colours in a colouring with a minimal number of colours

Graph concepts



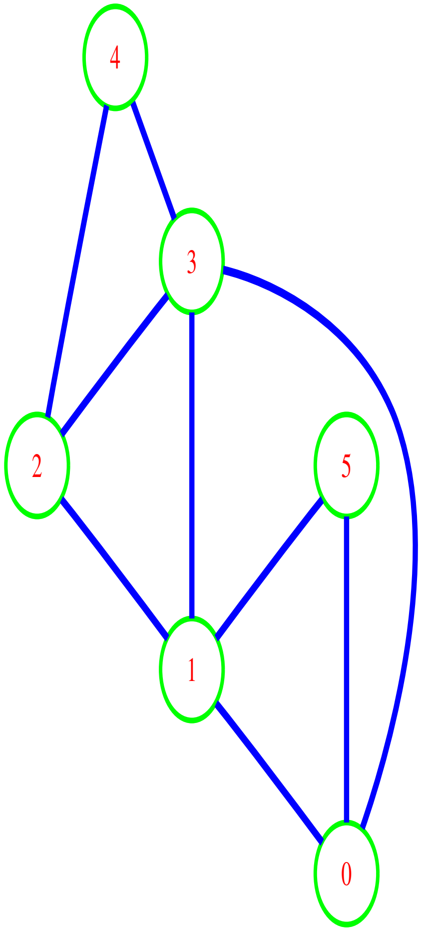
- *clique* - a complete subgraph
- *clique number* ω - the number of nodes in a largest clique
- *colouring* - an assignment of colours to nodes in which no neighbours have the same colour
- *chromatic number* χ - the number of colours in a colouring with a minimal number of colours
- *edge colouring* - an assignment of colours to edges in which no edges with a common endpoint have the same colour

Graph concepts



- *clique* - a complete subgraph
- *clique number* ω - the number of nodes in a largest clique
- *colouring* - an assignment of colours to nodes in which no neighbours have the same colour
- *chromatic number* χ - the number of colours in a colouring with a minimal number of colours
- *edge colouring* - an assignment of colours to edges in which no edges with a common endpoint have the same colour
- *edge chromatic number* χ' - the number of colours in an edge colouring with a minimal number of colours

Graph concepts



- *clique* - a complete subgraph
- *clique number* ω - the number of nodes in a largest clique
- *colouring* - an assignment of colours to nodes in which no neighbours have the same colour
- *chromatic number* χ - the number of colours in a colouring with a minimal number of colours
- *edge colouring* - an assignment of colours to edges in which no edges with a common endpoint have the same colour
- *edge chromatic number* χ' - the number of colours in an edge colouring with a minimal number of colours

Semidefinite programming and graph theory

Semidefinite programming (SDP) is a kind of generalized linear programming.

- Many practical optimization problems can be formulated as SDPs. There has been rapid progress in the last 20 years on understanding the theory of SDP, but the development of SDP software that is convenient to use has lagged behind.

Semidefinite programming and graph theory

Semidefinite programming (SDP) is a kind of generalized linear programming.

- Many practical optimization problems can be formulated as SDPs. There has been rapid progress in the last 20 years on understanding the theory of SDP, but the development of SDP software that is convenient to use has lagged behind.
- Good software is now available. This project would use sdpsol for small test problems, and then move to DSDP for large problems.

Semidefinite programming and graph theory

Semidefinite programming (SDP) is a kind of generalized linear programming.

- Many practical optimization problems can be formulated as SDPs. There has been rapid progress in the last 20 years on understanding the theory of SDP, but the development of SDP software that is convenient to use has lagged behind.
- Good software is now available. This project would use sdpsol for small test problems, and then move to DSDP for large problems.
- The aim of the project is to see what performance one can achieve on real graph theory problems of the types that come up in network modelling.

Semidefinite programming and graph theory

Semidefinite programming (SDP) is a kind of generalized linear programming.

- Many practical optimization problems can be formulated as SDPs. There has been rapid progress in the last 20 years on understanding the theory of SDP, but the development of SDP software that is convenient to use has lagged behind.
- Good software is now available. This project would use sdpsol for small test problems, and then move to DSDP for large problems.
- The aim of the project is to see what performance one can achieve on real graph theory problems of the types that come up in network modelling.
- Such problems include the Lovasz theta number (a lower bound for the chromatic number), the maximal stable set problem, and the maximum cut problem.

Semidefinite programming and graph theory

Semidefinite programming (SDP) is a kind of generalized linear programming.

- Many practical optimization problems can be formulated as SDPs. There has been rapid progress in the last 20 years on understanding the theory of SDP, but the development of SDP software that is convenient to use has lagged behind.
- Good software is now available. This project would use sdpsol for small test problems, and then move to DSDP for large problems.
- The aim of the project is to see what performance one can achieve on real graph theory problems of the types that come up in network modelling.
- Such problems include the Lovasz theta number (a lower bound for the chromatic number), the maximal stable set problem, and the maximum cut problem.
- An important outcome of the project would be a determination of how the computation time scales with problem size. Geometric optimization problems could also be studied.

2. Singular value decomposition updating

- The Singular value decomposition (SVD) of a matrix is a basic computation in numerical linear algebra, with many applications such as in statistics and signal processing.

2. Singular value decomposition updating

- The Singular value decomposition (SVD) of a matrix is a basic computation in numerical linear algebra, with many applications such as in statistics and signal processing.
- Updating the SVD usually means recomputing the factors after adding a row or column, and algorithms for this have been much studied.

2. Singular value decomposition updating

- The Singular value decomposition (SVD) of a matrix is a basic computation in numerical linear algebra, with many applications such as in statistics and signal processing.
- Updating the SVD usually means recomputing the factors after adding a row or column, and algorithms for this have been much studied.
- However, I have an application in signal processing, where only the approximate SVD is required, but it is required to rapidly update the factors after a small change in the matrix (perturbation theory, if you like).

2. Singular value decomposition updating

- The Singular value decomposition (SVD) of a matrix is a basic computation in numerical linear algebra, with many applications such as in statistics and signal processing.
- Updating the SVD usually means recomputing the factors after adding a row or column, and algorithms for this have been much studied.
- However, I have an application in signal processing, where only the approximate SVD is required, but it is required to rapidly update the factors after a small change in the matrix (perturbation theory, if you like).
- I have an idea how to do this, and the project would be to develop the idea and implement it in software.

3. Singular value decomposition methods for Lyapunov exponents

- Lyapunov exponents are basic quantities in dynamical systems theory; roughly speaking they quantify the average divergence of orbits.

3. Singular value decomposition methods for Lyapunov exponents

- Lyapunov exponents are basic quantities in dynamical systems theory; roughly speaking they quantify the average divergence of orbits.
- Standard methods use QR or SVD decompositions, and need the local linearization of the flow.

3. Singular value decomposition methods for Lyapunov exponents

- Lyapunov exponents are basic quantities in dynamical systems theory; roughly speaking they quantify the average divergence of orbits.
- Standard methods use QR or SVD decompositions, and need the local linearization of the flow.
- However, a method proposed by Dieci and Elia shows how to get the singular values much more directly, without performing a full SVD on each time-step.

3. Singular value decomposition methods for Lyapunov exponents

- Lyapunov exponents are basic quantities in dynamical systems theory; roughly speaking they quantify the average divergence of orbits.
- Standard methods use QR or SVD decompositions, and need the local linearization of the flow.
- However, a method proposed by Dieci and Elia shows how to get the singular values much more directly, without performing a full SVD on each time-step.
- This project would improve on this by using automatic differentiation (AD) techniques to compute the local linearizations.

3. Singular value decomposition methods for Lyapunov exponents

- Lyapunov exponents are basic quantities in dynamical systems theory; roughly speaking they quantify the average divergence of orbits.
- Standard methods use QR or SVD decompositions, and need the local linearization of the flow.
- However, a method proposed by Dieci and Elia shows how to get the singular values much more directly, without performing a full SVD on each time-step.
- This project would improve on this by using automatic differentiation (AD) techniques to compute the local linearizations.
- I have much experience in AD and can supply software for this step.

4. Automatic differentiation (AD) methods for ODE sensitivity analysis

- ODEs are routinely solved numerically by Runge-Kutta and similar methods. but often we would like to know the derivatives with respect to initial values and with respect to parameters, of the solution.

4. Automatic differentiation (AD) methods for ODE sensitivity analysis

- ODEs are routinely solved numerically by Runge-Kutta and similar methods. but often we would like to know the derivatives with respect to initial values and with respect to parameters, of the solution.
- This means solving another linear system of ODEs along the solution. It can be tedious to calculate this linear system by hand, but fortunately it is possible to automate this with AD techniques.

4. Automatic differentiation (AD) methods for ODE sensitivity analysis

- ODEs are routinely solved numerically by Runge-Kutta and similar methods. but often we would like to know the derivatives with respect to initial values and with respect to parameters, of the solution.
- This means solving another linear system of ODEs along the solution. It can be tedious to calculate this linear system by hand, but fortunately it is possible to automate this with AD techniques.
- This project would develop software for this purpose and evaluate its efficiency.

4. Automatic differentiation (AD) methods for ODE sensitivity analysis

- ODEs are routinely solved numerically by Runge-Kutta and similar methods. but often we would like to know the derivatives with respect to initial values and with respect to parameters, of the solution.
- This means solving another linear system of ODEs along the solution. It can be tedious to calculate this linear system by hand, but fortunately it is possible to automate this with AD techniques.
- This project would develop software for this purpose and evaluate its efficiency.
- C++ programming is needed (as operator overloading is necessary).

5. Fast random selection

- Consider the problem: a large number N of objects are presented to us one by one, and we wish to either:
  - *Select exactly some specified number* $n \leqslant N$ *of them*
  - *Select each with some specified probability* p

5. Fast random selection

- Consider the problem: a large number N of objects are presented to us one by one, and we wish to either:
  - *Select exactly some specified number* $n \leqslant N$ *of them*
  - *Select each with some specified probability* p

- The problem becomes hard to solve efficiently when p is small, because to generate a random number which usually results in a rejection is inefficient. It would be better to skip over a block of items. Ways to do these were proposed by Vitter in ACM Trans Math Soft 13, 58 (1987).

5. Fast random selection

- Consider the problem: a large number N of objects are presented to us one by one, and we wish to either:
  - *Select exactly some specified number $n \leqslant N$ of them*
  - *Select each with some specified probability p*

- The problem becomes hard to solve efficiently when p is small, because to generate a random number which usually results in a rejection is inefficient. It would be better to skip over a block of items. Ways to do these were proposed by Vitter in ACM Trans Math Soft 13, 58 (1987).
- This project would investigate these methods and check their efficiency in practice. There are applications to the generation of large random graphs.

6. Random sampling of set partitions

- A partition of $[n] = \{1, 2, ..., n\}$ is an assignment each element to a subset called a block, without regard to the labelling of the block, or order of elements within a block. Thus for [5], 01|2|34 and 2|10|43 are the same partition.

6. Random sampling of set partitions

- A partition of $[n] = \{1, 2, ..., n\}$ is an assignment each element to a subset called a block, without regard to the labelling of the block, or order of elements within a block. Thus for [5], 01|2|34 and 2|10|43 are the same partition.
- If there are k blocks, we speak of a k-partition. The number of k-partitions of $[n]$ is given by the Stirling number of the second kind $S(n, k)$

6. Random sampling of set partitions

- A partition of $[n] = \{1, 2, ..., n\}$ is an assignment each element to a subset called a block, without regard to the labelling of the block, or order of elements within a block. Thus for [5], 01|2|34 and 2|10|43 are the same partition.
- If there are k blocks, we speak of a k-partition. The number of k-partitions of $[n]$ is given by the Stirling number of the second kind $S(n, k)$
- The total number of partitions of $[n]$ is given by the sum over k of $S(n, k)$, and this is called a Bell number $B(k)$.

6. Random sampling of set partitions

- A partition of $[n] = \{1, 2, ..., n\}$ is an assignment each element to a subset called a block, without regard to the labelling of the block, or order of elements within a block. Thus for [5], 01|2|34 and 2|10|43 are the same partition.
- If there are k blocks, we speak of a k-partition. The number of k-partitions of $[n]$ is given by the Stirling number of the second kind $S(n, k)$
- The total number of partitions of $[n]$ is given by the sum over k of $S(n, k)$, and this is called a Bell number $B(k)$.
- There are algorithms for generating all partitions in a Gray code order, due to Ehrlich, Ruskey. In such a sequence, only one element moves to a new block on each step.

6. Random sampling of set partitions

- A partition of $[n] = \{1, 2, ..., n\}$ is an assignment each element to a subset called a block, without regard to the labelling of the block, or order of elements within a block. Thus for [5], 01|2|34 and 2|10|43 are the same partition.
- If there are k blocks, we speak of a k-partition. The number of k-partitions of $[n]$ is given by the Stirling number of the second kind $S(n, k)$
- The total number of partitions of $[n]$ is given by the sum over k of $S(n, k)$, and this is called a Bell number $B(k)$.
- There are algorithms for generating all partitions in a Gray code order, due to Ehrlich, Ruskey. In such a sequence, only one element moves to a new block on each step.
- Beyond about $n = 15$, there are too many partitions to allow us to construct all of them in reasonable time.

6. Random sampling of set partitions

- A partition of $[n] = \{1, 2, ..., n\}$ is an assignment each element to a subset called a block, without regard to the labelling of the block, or order of elements within a block. Thus for [5], 01|2|34 and 2|10|43 are the same partition.
- If there are k blocks, we speak of a k-partition. The number of k-partitions of $[n]$ is given by the Stirling number of the second kind $S(n, k)$
- The total number of partitions of $[n]$ is given by the sum over k of $S(n, k)$, and this is called a Bell number $B(k)$.
- There are algorithms for generating all partitions in a Gray code order, due to Ehrlich, Ruskey. In such a sequence, only one element moves to a new block on each step.
- Beyond about $n = 15$, there are too many partitions to allow us to construct all of them in reasonable time.
- Thus, for large sets, simulations must rely on a uniform random sampling procedure. This project would investigate such methods and produce a well-designed C library for use in other projects.

7. Fast counting

- This project is inspired by the paper *HyerLoglog: the analysis of a near-optimal cardinality estimation algorithm* of Flajolet et al.

7. Fast counting

- This project is inspired by the paper *HyerLoglog: the analysis of a near-optimal cardinality estimation algorithm* of Flajolet et al.
- How do we estimate (rapidly, and with minimal storage) the number of distinct items in a very large set (or data stream)? In other words, the problem concerns estimating the cardinality of a multiset.

7. Fast counting

- This project is inspired by the paper *HyerLoglog: the analysis of a near-optimal cardinality estimation algorithm* of Flajolet et al.
- How do we estimate (rapidly, and with minimal storage) the number of distinct items in a very large set (or data stream)? In other words, the problem concerns estimating the cardinality of a multiset.
- Flajolet et al. previously developed their loglog algorithm, and I worked on an efficient C implementation of this. They have now improved this with the hyperloglog algorithm, and this project would be to implement this and compare its performance in practice with alternatives.

7. Fast counting

- This project is inspired by the paper *HyerLoglog: the analysis of a near-optimal cardinality estimation algorithm* of Flajolet et al.
- How do we estimate (rapidly, and with minimal storage) the number of distinct items in a very large set (or data stream)? In other words, the problem concerns estimating the cardinality of a multiset.
- Flajolet et al. previously developed their loglog algorithm, and I worked on an efficient C implementation of this. They have now improved this with the hyperloglog algorithm, and this project would be to implement this and compare its performance in practice with alternatives.
- The algorithm should have practical applications in informatics; for example, counting the number of different packet types in a network.

8. Random sampling of unlabelled structures

- This project is inspired by the paper *Boltzmann Sampling of Unlabelled Structures* of Flajolet et al.

8. Random sampling of unlabelled structures

- This project is inspired by the paper *Boltzmann Sampling of Unlabelled Structures* of Flajolet et al.
- I quote from their abstract: "Boltzmann models from statistical physics, combined with methods from analytic combinatorics, give rise to efficient algorithms for the random generation of unlabelled objects.

8. Random sampling of unlabelled structures

- This project is inspired by the paper *Boltzmann Sampling of Unlabelled Structures* of Flajolet et al.
- I quote from their abstract: "Boltzmann models from statistical physics, combined with methods from analytic combinatorics, give rise to efficient algorithms for the random generation of unlabelled objects.
- The resulting algorithms generate in an unbiased manner discrete configurations that may have nontrivial symmetries, and they do so by means of real-arithmetic computations.

8. Random sampling of unlabelled structures

- This project is inspired by the paper *Boltzmann Sampling of Unlabelled Structures* of Flajolet et al.
- I quote from their abstract: "Boltzmann models from statistical physics, combined with methods from analytic combinatorics, give rise to efficient algorithms for the random generation of unlabelled objects.
- The resulting algorithms generate in an unbiased manner discrete configurations that may have nontrivial symmetries, and they do so by means of real-arithmetic computations.
- This project would develop software for some of these methods, and measure its efficiency in practice. There are important applications in statistical computing.

9. Convex optimization in python

- Convex optimization is very important in many application areas, and is also theoretically very elegant.

9. Convex optimization in python

- Convex optimization is very important in many application areas, and is also theoretically very elegant.
- But these techniques are not used as often as should be in practice because of the lack of software which satisfies all the requirements of being free and open-source (which rules out matlab-based software), easy-to-use, powerful, and well designed. This seems to be changing with the python-based CVXOPT and CVXMOD.

9. Convex optimization in python

- Convex optimization is very important in many application areas, and is also theoretically very elegant.
- But these techniques are not used as often as should be in practice because of the lack of software which satisfies all the requirements of being free and open-source (which rules out matlab-based software), easy-to-use, powerful, and well designed. This seems to be changing with the python-based CVXOPT and CVXMOD.
- This project would survey the field of convex optimization and investigate and evaluate this software. We would hope to try applications in the field of radio technology.

10. Geographical computations

- In this project, I developed an algorithm for computing intervisibility of points on the earth's surface, using elevation data from a shuttle flight.

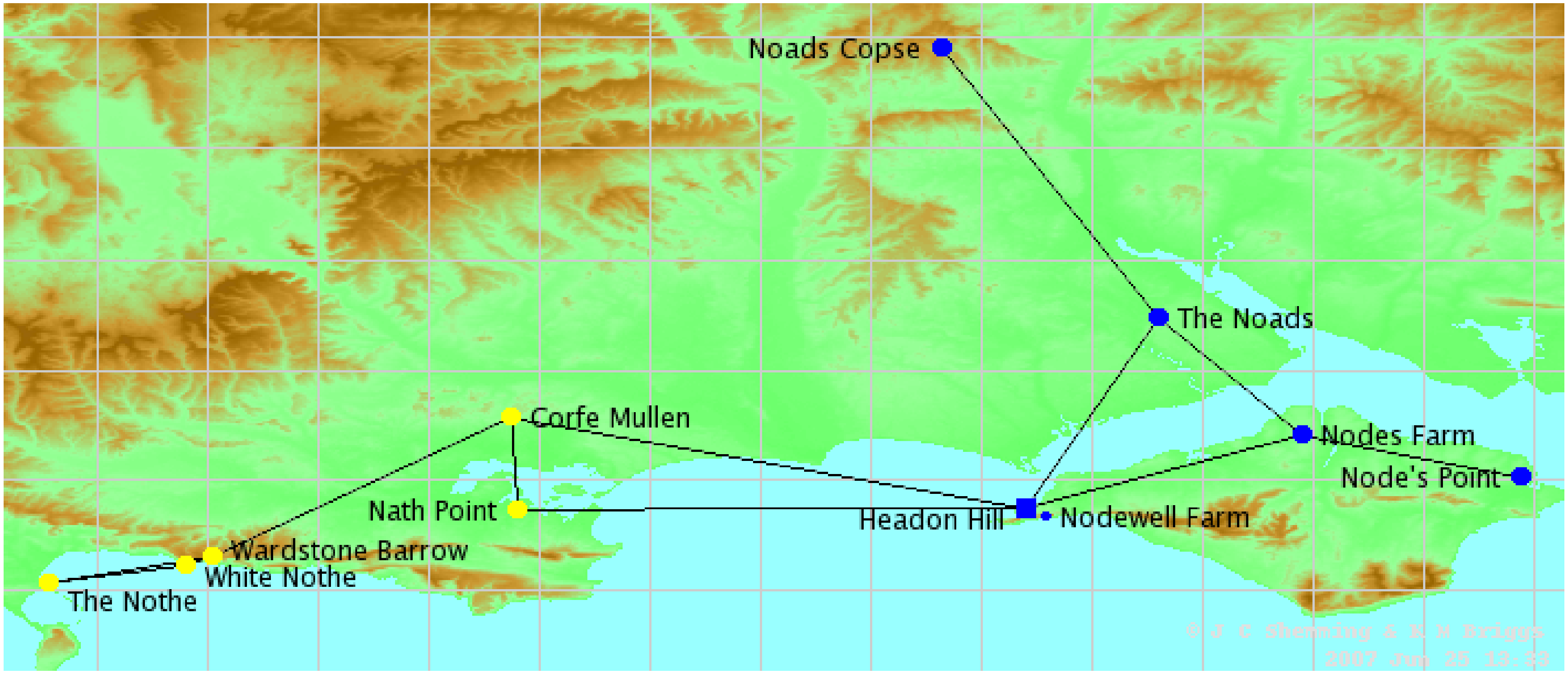
10. Geographical computations

- In this project, I developed an algorithm for computing intervisibility of points on the earth's surface, using elevation data from a shuttle flight.
- This project would take this further, to compute the entire region visible from a given point. Computer graphics would be used to draw the region. There is actually some real maths here - we have to handle nonlinear 3d coordinate systems.

10. Geographical computations

- In this project, I developed an algorithm for computing intervisibility of points on the earth's surface, using elevation data from a shuttle flight.
- This project would take this further, to compute the entire region visible from a given point. Computer graphics would be used to draw the region. There is actually some real maths here - we have to handle nonlinear 3d coordinate systems.
- The application is to the historical study of ancient defence systems.

10. Geographical computations ctnd.



11. Statistics of Roman roads in Britain

- The Romans built roads all over Britain, and I would like some statistical characterization of the network - what is the diameter of the graph, what is the distribution of distances to the nearest road, and so on.

11. Statistics of Roman roads in Britain

- The Romans built roads all over Britain, and I would like some statistical characterization of the network - what is the diameter of the graph, what is the distribution of distances to the nearest road, and so on.
- I have a database, and a major part of this project would be to classify all the roads, probably as major, minor and doubtful. We would want separate statistical information for each category.

11. Statistics of Roman roads in Britain

- The Romans built roads all over Britain, and I would like some statistical characterization of the network - what is the diameter of the graph, what is the distribution of distances to the nearest road, and so on.
- I have a database, and a major part of this project would be to classify all the roads, probably as major, minor and doubtful. We would want separate statistical information for each category.
- The maths involved is interesting computational geometry, and fast algorithms are needed for things like the point-in-polygon test, and for distance to the nearest of a given finite set of straight line segments.

11. Statistics of Roman roads in Britain ctnd.