Theory of Real Approximation and Exact Geometric Computation

Chee Yap
Courant Institute of Mathematical Sciences
Department of Computer Science
New York University

This talk is produced using the ppower4 software.
The "pause/highlight" feature used in the original talk slides has been turned off for this version of the talk.

ABSTRACT

Exact Geometric Computation (EGC) has been developed over the last decade by computational geometers in response to the widespread problem of numerical non-robustness in geometric algorithms. It is currently the most successful approach to nonrobustness, and its technology has been encoded in libraries such as LEDA, CGAL and Core Library. The unique requirement of EGC is the necessity to decide zero in its computation. The zero problem arises in many challenging problems such as surface intersection in geometric modeling, meshing of implicit surfaces with guaranteed topology, determination of singularity and automated geometric theorem proving. The complexity of such zero problems ranges from easy to undecidable. We seek a theory of computation that properly accounts for this phenomenon.

Standard computability (via Turing machines) is a theory of computation over countable domains such as strings or natural numbers. Computing over uncountable domains such as real numbers turns out to be much more subtle. There are two current approaches in the theory of real computation: the analytic model going back to Turing (1936) and Grzegorcyzk (1955), and the algebraic model represented by the Blum-Shub-Smale model and Real RAMs of theoretical computer science. Smale and others have pointed out that such a theory amounts to a foundation for scientific computation and numerical analysis. Unfortunately the zero problem is undecidable in the analytic model and trivial in the algebraic model.

We propose a synthesis to resolve this discrepancy. First, we modify the analytic approach so that the primary focus is on approximability rather than computability. The zero problem is embedded in the problem of relative approximability of functions, and the central open questions revolves around the composability of functions. We modify Schoenhage's pointer machine model to capture geometric computation, which is characterized by combinatorial structures implicitly defined by numerical data. The algebraic model reappears in our synthesis as an abstract computation model for which the central questions revolves around transfer theorems: when is algebraic computability imply approximability?

In this talk, we first give a brief introduction to EGC and survey some recent results on zero problems. We outline the theory of real approximation and show its relation to analytic computability and algebraic computability.

TALK OVERVIEW

- What is Exact Geometric Computation (EGC)?
- Why care about nothing (Zero)?
- What can be Solved in the EGC sense?
- A New Synthesis: Theory of Real Approximation

PART I.
WHAT IS EGC?

Nonrobustness Phenomenon

- Software Loops, Crashes or Produces Inconsistent Output
  - caused by numerical errors
  - intermittent
- E.g., Mesh Generation
  - Point Classification Problem
  - Misclassification $\Rightarrow$ Dirty meshes
- E.g., Boolean Operations on Sets
  - Example next
- Etc, etc.

Nonrobust Commercial CAD Software

- From Mehlhorn and Hart (2001)

SYSTEM	n	α	TIME	OUTPUT
ACIS	1000	1.0e-4	5 min	correct
ACIS	1000	1.0e-5	4.5 min	correct
ACIS	1000	1.0e-6	30 min	too difficult!
Microstation95	100	1.0e-2	2 sec	correct
Microstation95	100	0.5e-2	3 sec	incorrect!
Rhino3D	200	1.0e-2	15 sec	correct
Rhino3D	400	1.0e-2	–	crash!
CGAL/LEDA	5000	6.175e-6	30 sec	correct
CGAL/LEDA	5000	1.581e-9	34 sec	correct
CGAL/LEDA	20000	9.88e-7	141 sec	correct

  - Last three rows, from the CGAL/LEDA software, are always correct.
  - This last column shows 3 *forms of failure*.

Example of CFD in Aircraft Industry

Computational Fluid Dynamics (CFD) applications with 50 million elements
– quote from T. Peters and D. Ferguson (Boeing Company)

- 10-20 minutes for surface mesh generation
- 3-4 hours for volume meshing
- 1 hour for actual flow analysis
- 2-4 weeks for geometry repair

Techniques now exists to eliminate the repair stage...

The Economic Imperative: why we must solve this

- Negative Economic Impact
  - Nonrobustness is barrier to full automation in industry
  - Software bugs cost billions of $/year to US industries (NIST Report 2002)
- Scientific Productivity
  - Wasted Programmers/researchers' time
- Mission Critical Computation
  - Patriot missile in Gulf War (1990)
  - French Ariane Rocket failure (1995)
  - North Sea Oil Platform Collapse (1996)
  - ...

Paths to Robust Numerical Algorithms

- In practice: Epsilon-Tweaking
  - Never compare against zero
  - IF $(x < |\varepsilon|)$ THEN ... ELSE ...

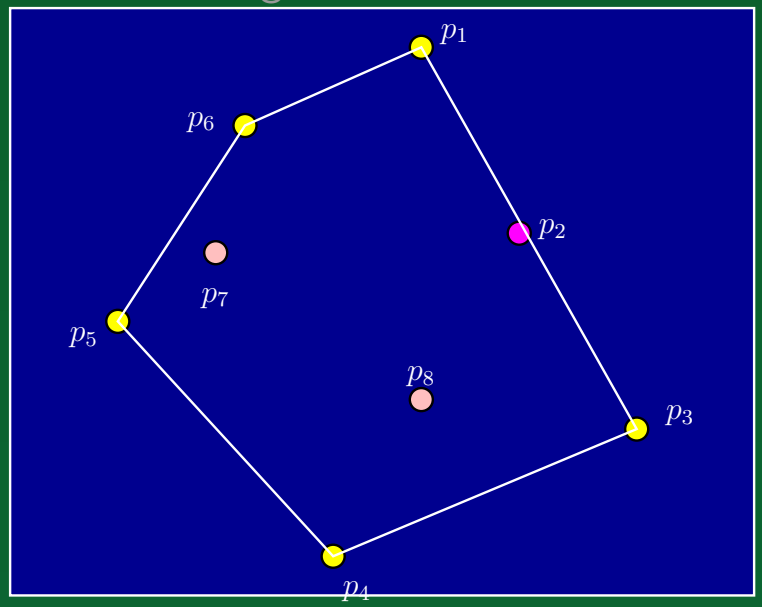
- Approaches in the Computational Geometry literature:
  - Numerically Stable Algorithms (Fortune,...)
  - Finite Precision Geometry (Yao-Greene, Sugihara,...)
  - Topological Approach (Sugihara-Iri, Milenkovic,...)
  - Consistency Approach (Hopcroft-Hoffman-Karasick, Fortune,...)
  - Interval Geometry Approach (Guibas, Sequin-Segal,...)
  - Exact Geometric Computation

Again, What is Geometry?

- Euclid, Descartes, Klein, Hilbert, Tarski, Dieudonné, ...
  - Alexandrov, Toth, Atiyah, Chern, ...
- Computational Perspective: Geometry is about discrete relations
  - Is a point on a line?
  - Does a plane intersect a sphere?
- "Geometry = Combinatorics + Numerics"
  - E.g., 2-D Convex Hull = circular graph + point labels

Again, What is Geometry?

- Euclid, Descartes, Klein, Hilbert, Tarski, Dieudonné, ...
  - Alexandrov, Toth, Atiyah, Chern, ...
- Computational Perspective: Geometry is about discrete relations
  - Is a point on a line?
  - Does a plane intersect a sphere?
- "Geometry = Combinatorics + Numerics"
  - E.g., 2-D Convex Hull = circular graph + point labels



- Implicit Consistency Requirement
  - E.g., Points of convex hull in convex position
  - If not, we get a "qualitative" error

How to Compute Exactly, in the Geometric Sense

- Algorithm Execution = Sequence of Steps
- Step = either Construction Step or Test Step
  - Construction Step: $x := x + 2$
  - Test Step: IF $(x = 0)$ THEN . . . ELSE . . .
- Test Step = determines the path in computation tree
- Combinatorial relations are determined by the path taken
- THEREFORE, if all comparisons are correct, we take the correct path
  - HENCE, the geometry will be correct
  - *This is the take home slide!*

Features of EGC

- "Exactness" is in the geometry, NOT the arithmetic
- Approximate numbers CAN/MUST be exploited
  - Floating point filters
- ECG uses standard geometry (!)
- ECG is "algorithm-neutral"
  - Any correct algorithm can be used (even "unstable ones")
  - No need to design special "robust algorithms"
- Running time has adaptive complexity
  - Related to condition numbers
- Practical experience from LEDA and CGAL
  - Factor of 2-10 for basic 2-D and 3-D problems
  - Significant user base, including industrial partners
- Most successful approach to nonrobustness

Mini-summary, Part I

- Solving numerical nonrobustness is an economic imperative
  - "Our little closet secret" – no one talks about it, but everyone knows it is there
- Exact Geometric Computation provides a systematic solution
  - Widely applicable through EGC number libraries

PART II. Why Care about Nothing (Zero)?

"The history of the zero recognition problem is somewhat confused by the fact that many people do not recognize it as a problem at all."

— Daniel Richardson (1996)

Why Care About Exactness?

- My input is inexact
  - Treat input as (nominally) exact
  - Reformulate inexact problem as an exact one – back to EGC
- What about Consistency without Exactness?
  - Too hard [Hopcroft-Hoffman-Karasick, Fortune]
  - EGC is the simplest way to achieve consistency!
- Backward Error Analysis
  - Numerical Analysis Paradigm for Inexact Problems
  - Breaks down for Seminumerical Problems
  - Inexact solution may be no easier than EGC
- Deciding Zero may be inevitable
  - Needs no convincing in this audience?

An Object Lesson

- But is it REALLY zero?

$$\begin{aligned}\sqrt{2}+\sqrt{3}-\sqrt{5+2\sqrt{6}} &= 1.4142+1.7320-\sqrt{5+2\times 2.4494}\\ &= 3.1462-\sqrt{9.8989}\\ &= 3.1462-3.1462\\ &= 0?\end{aligned}$$

- Paper of Ed Scheinerman, "How Close is Close enough?"
  - Amer.Math.Monthly, 107(2000)489–499.
- Core Demo
  - What was the lesson the Teacher was trying to convey?
  - Is the Teacher right?

What Appears to be the Problem?

- Is a number equal to zero?
  - Decision Problem – YES/NO answers
- Why is any effort needed at all?
  - Numbers have canonical names
  - E.g., zero, one, two, half, negative ten, square-root two, pi, etc
  - In symbols, 0, 1, 2, $\frac{1}{2}$, -10, $\sqrt{2}$, π, $\ldots$
- Numerical Expressions (non-canonical)
  - $1 - 1 + 1 - 1$,
  - $2^2 + 5 - 3^2$,
  - $1 - \sum_{n=1}^{\infty} 2^{-n}$,
  - $\sqrt{2} + \sqrt{3} - \sqrt{5 + 2\sqrt{6}}$
  - These are all 0

Apparent Zeros

- Folklore:
  - $3 \cdot \log(640320)/\sqrt{163} - \pi \ < 10^{-15}$
- Graham:
  - $$\begin{aligned} & \left(\sqrt{1000001} + \sqrt{1000025} + \sqrt{1000031} + \sqrt{1000084} + \sqrt{1000087}\right. \\ & \qquad \left. + \sqrt{1000134} + \sqrt{1000158} + \sqrt{1000182} + \sqrt{1000198}\right) \\ - \quad & \left(\sqrt{1000002} + \sqrt{1000018} + \sqrt{1000042} + \sqrt{1000066} + \sqrt{1000113}\right. \\ & \qquad \left. + \sqrt{1000116} + \sqrt{1000169} + \sqrt{1000175} + \sqrt{1000199}\right) \\ < \quad & 10^{-36} \end{aligned}$$
- Richardson:
  - $F(F(10^{-126})) \ < 10^{-1141}$
  - where $F(x) = (1+x)^{1/2} - 2(1+3x/4)^{1/3} + 1$

The Numerical Halting Problem

- Equality Decision: If $x = y$?
  - Compute approximations $\widetilde{x}, \widetilde{y}$ to 1002 bits
  - If $|\widetilde{x} - \widetilde{y}| < 2^{-1000}$, conclude that $x = y$
  - If not, compute to 2002 bits, then 4002 bits, etc
  - If $x = y$, we cannot halt!

- Assume: Can evaluate x to any precision (absolute or relative)

- Suppose we have an *a priori* bound $B = B(x, y) > 0$
  - such that, if $x \neq y$, then $|x - y| > B$
  - This provides the stopping criteria !
  - B is called a (conditional) zero bound

- Feature of Zero Bound approach
  - For algebraic expressions, such bounds are possible
  - Semi-adaptivity — if $x \neq y$, the complexity depends on $-\log_2 |x - y|$

The Zero Problems

- Let us formalize this...
- Let Ω be a set of numerical operators
  - Let $E(\Omega)$ denote expressions over Ω
  - E.g., if $\Omega = \{+, -, \times, \div\} \cup \mathbb{Z}$ then $e = (2 + (1 - 3)) \div (1 - 1)$ is an expression in $E(\Omega)$
- Evaluation Function: $val : E(\Omega) \dashrightarrow \mathbb{C}$
- The Zero Problem for Ω, $ZERO(\Omega)$, is this:
  - Given $e \in E(\Omega)$, is $val(e) = 0$?
- Key Open Problem:
  - Is the Zero Problem for $\Omega = \{\pm, \times, \div, \sqrt[n]{\cdot}, \exp, \log\} \cup \mathbb{Z}$ decidable?
  - Richardson (1997): Decidable if Schanuel's Conjecture holds
- A Zero Problem is a "canonical expression" problem in Computer Algebra

Robust Computation for Every Programmer?

- The principles of EGC can be encoded into a general library
  - 1990's – handcoded EGC solutions
  - Vision: Any C++ Program can be robustified by calling an EGC library
- Two EGC Libraries
  - LEDA
  - Core Library
  - cf. Computer Algebra libraries, Real Libraries (iRRAM, MmxLib, MPFR, etc.)
- CORE Accuracy Levels:
  - Level I: IEEE Arithmetic
  - Level II: Arbitrary Accuracy
    - "Compute each operation to 1000 bits accuracy"
  - Level III: Guaranteed Accuracy
    - "Guarantee each variable has 1000 bits accuracy"

Core Library

- Delivery Mechanism in C++:

```
#define Core_Level 3
#include "CORE.h"

...  standard C++ Program here ...
```

- Every C++ program can be compiled into three distinct executable code!

- What is under the hood of CORE (and LEDA Real)?
  - $a = b + c$ constructs an Expression (DAG)
  - Can approximate to any relative or absolute precision

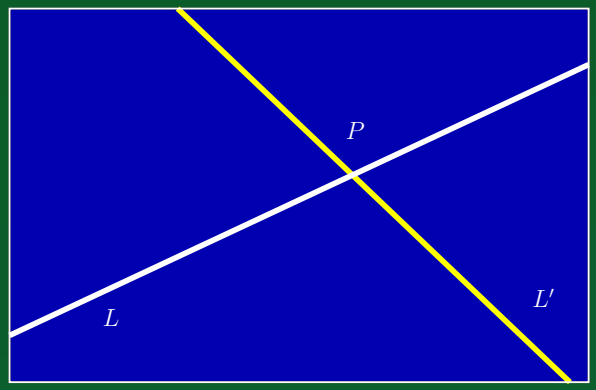
Core Demo

- A simple geometric test

  - Test if point P belongs to line L
- Core Demo
  - Intersect 1024 pairs of lines with a fixed plane P in space
  - Test if the intersection points lie in P
  - Result: In Level 1 accuracy, only 102 points lie on P
  - Result: In Level 3 accuracy, every point lies on P
- Knowing zero (and sign) yields correct geometry

Core Demo

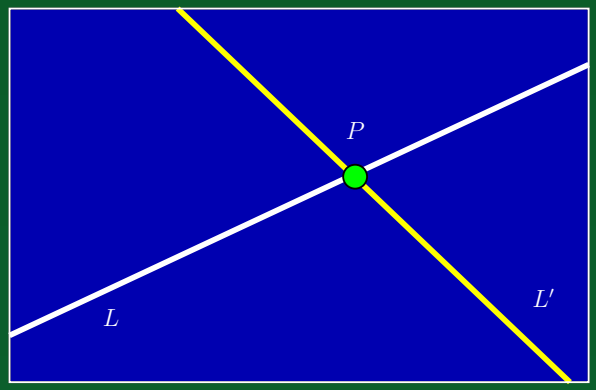
- A simple geometric test



  - Test if point P belongs to line L
- Core Demo
  - Intersect 1024 pairs of lines with a fixed plane P in space
  - Test if the intersection points lie in P
  - Result: In Level 1 accuracy, only 102 points lie on P
  - Result: In Level 3 accuracy, every point lies on P
- Knowing zero (and sign) yields correct geometry

Core Demo

- A simple geometric test



  - Test if point P belongs to line L
- Core Demo
  - Intersect 1024 pairs of lines with a fixed plane P in space
  - Test if the intersection points lie in P
  - Result: In Level 1 accuracy, only 102 points lie on P
  - Result: In Level 3 accuracy, every point lies on P
- Knowing zero (and sign) yields correct geometry

Mini-summary, Part II

- Central problem of EGC is the Zero Problem
  - Zero decision is often unavoidable
- Practical solutions must be semi-adaptive
  - based on theory of Zero Bounds
- Technology of Robust Computation
  - can now be made widely available, e.g., Core Library

PART III. WHAT CAN BE SOLVED IN THE EGC SENSE?

"It can be of no practical use to know that π is irrational, but if we can know, it surely would be intolerable not to know."

— E.C. Titchmarsh

Challenges in EGC Theory

- Solving nonlinear algebraic problems efficiently
  - E.g., Fully adaptive surface-surface intersection
- Better constructive zero bounds
- Development and generalizations of filters
  - Exploit fast machine arithmetic
- Degeneracy and perturbation techniques
- Compiler techniques in EGC software
- Geometric rounding – wide open
- Which problems be computed in the EGC sense?
  - (next)

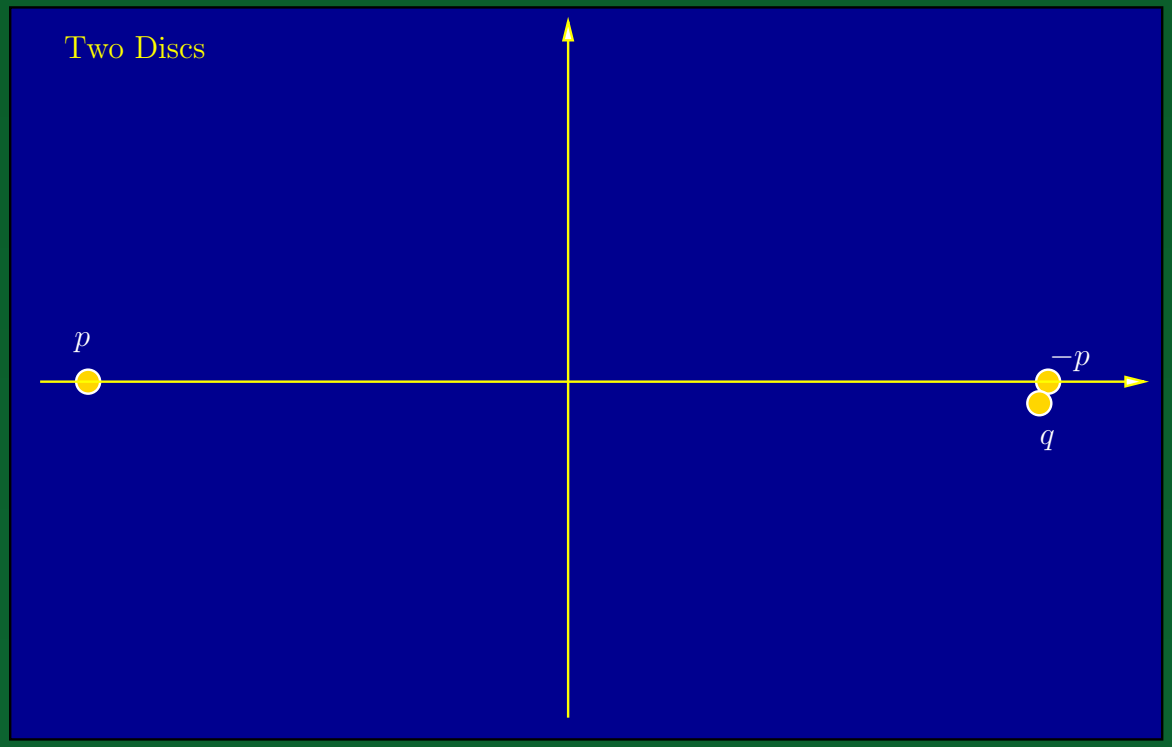
Shortest Path Amidst Disc Obstacles

- Given: Points $p, q \in \mathbb{R}^2$ and a collection S of discs
- Find: shortest path from p to q which avoids the obstacles in S
- 

- Dijkstra's shortest path algorithm: $O(n^2 \log n)$

Shortest Path Amidst Disc Obstacles

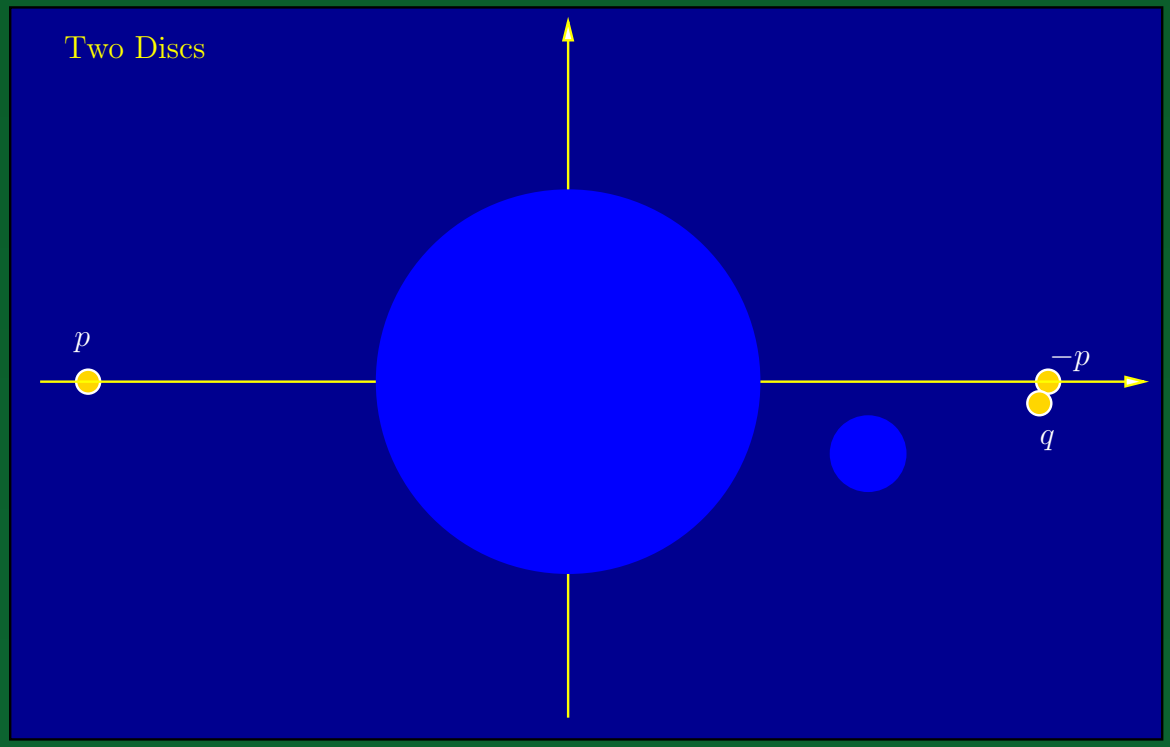
- Given: Points $p, q \in \mathbb{R}^2$ and a collection S of discs
- Find: shortest path from p to q which avoids the obstacles in S



- Dijkstra's shortest path algorithm: $O(n^2 \log n)$

Shortest Path Amidst Disc Obstacles

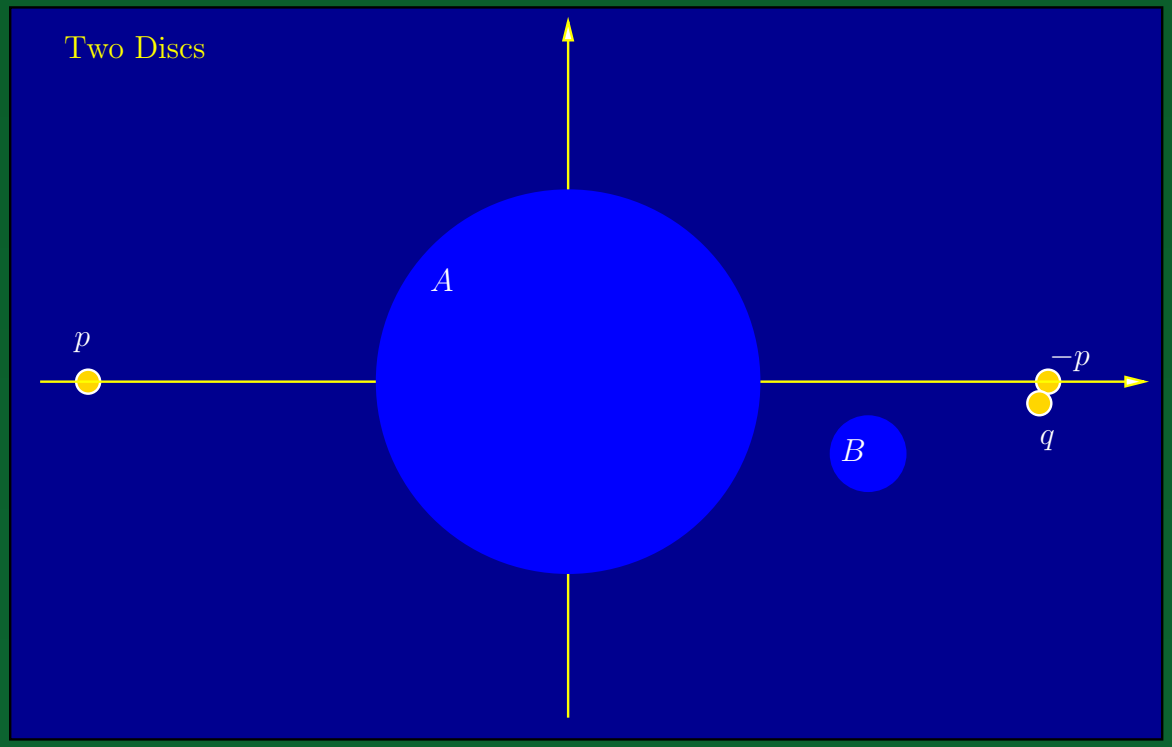
- Given: Points $p, q \in \mathbb{R}^2$ and a collection S of discs
- Find: shortest path from p to q which avoids the obstacles in S



- Dijkstra's shortest path algorithm: $O(n^2 \log n)$

Shortest Path Amidst Disc Obstacles

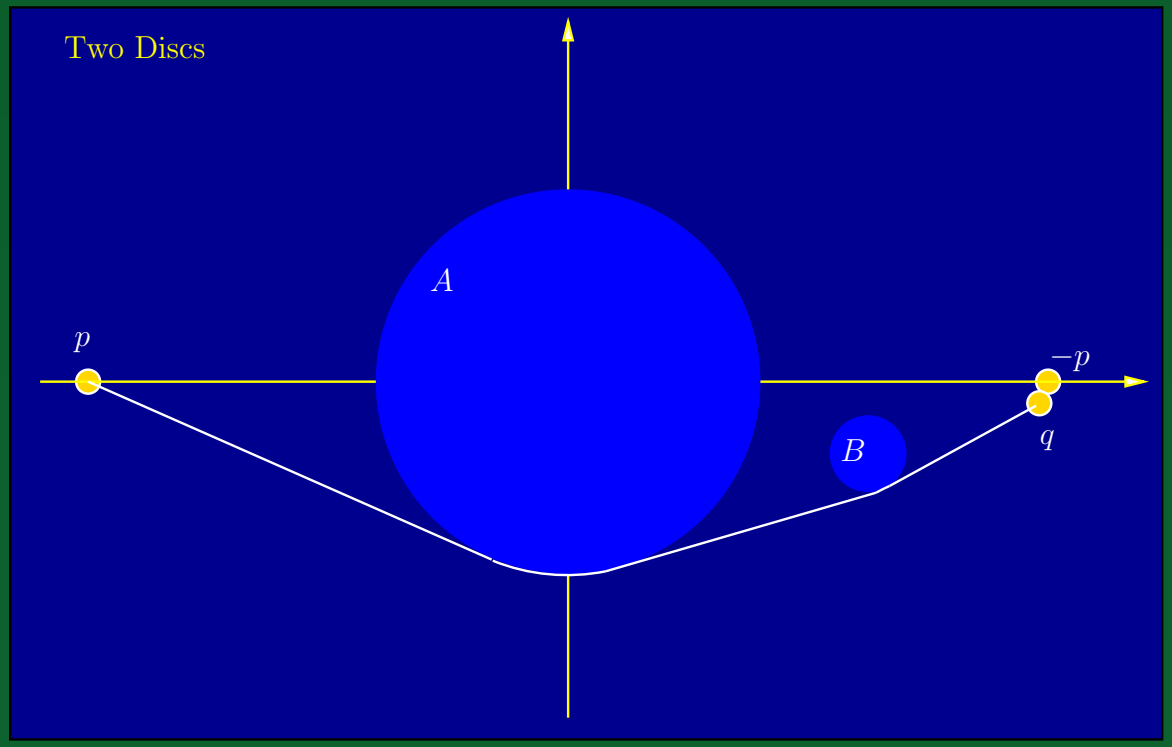
- Given: Points $p, q \in \mathbb{R}^2$ and a collection S of discs
- Find: shortest path from p to q which avoids the obstacles in S



- Dijkstra's shortest path algorithm: $O(n^2 \log n)$

Shortest Path Amidst Disc Obstacles

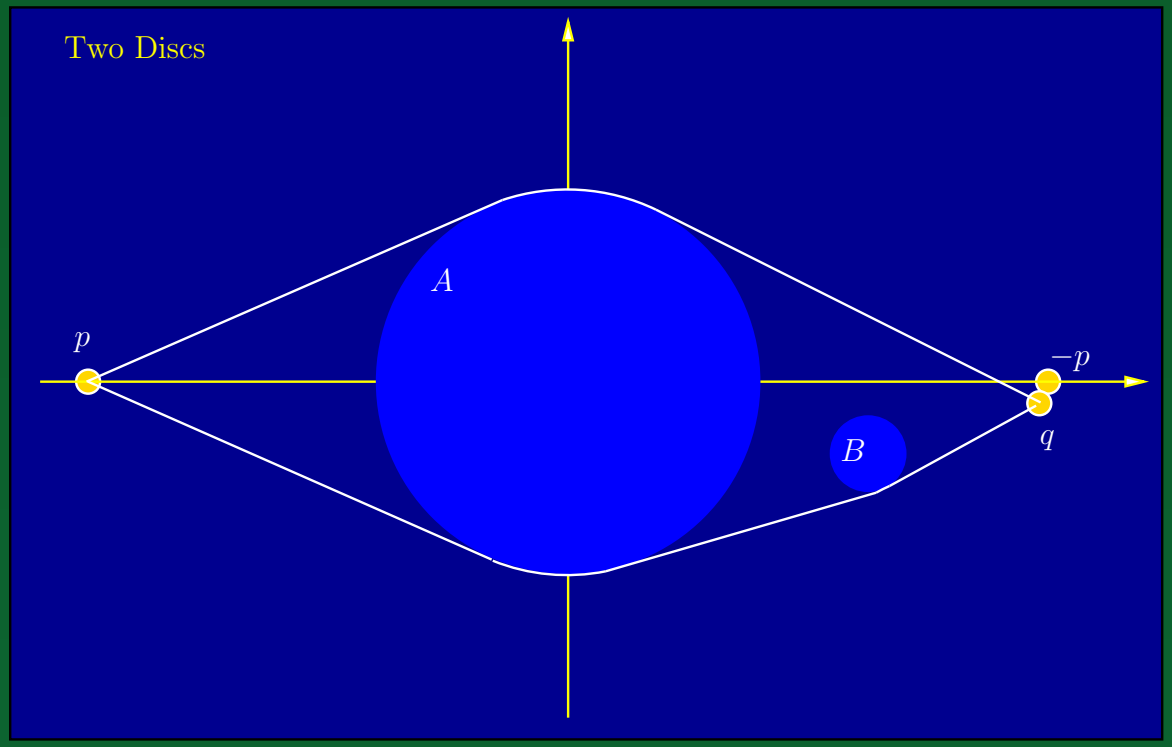
- Given: Points $p, q \in \mathbb{R}^2$ and a collection S of discs
- Find: shortest path from p to q which avoids the obstacles in S



- Dijkstra's shortest path algorithm: $O(n^2 \log n)$

Shortest Path Amidst Disc Obstacles

- Given: Points $p, q \in \mathbb{R}^2$ and a collection S of discs
- Find: shortest path from p to q which avoids the obstacles in S



- Dijkstra's shortest path algorithm: $O(n^2 \log n)$

What is Wrong with Dijkstra's Algorithm?

- Real RAM model assumed
- Length of a feasible path is

$$d(\mu) = \sum_{i=1}^{k} d(\mu_i) = \alpha + \sum_{i=1}^{m} \theta_i r_i \qquad (1)$$

  - $\alpha \geq 0$ is algebraic
  - $0 < r_1 < \cdots < r_m$ are distinct radii of discs
  - θ_i is total angle (in radians) around discs of radii r_i

Is it really Transcendental?

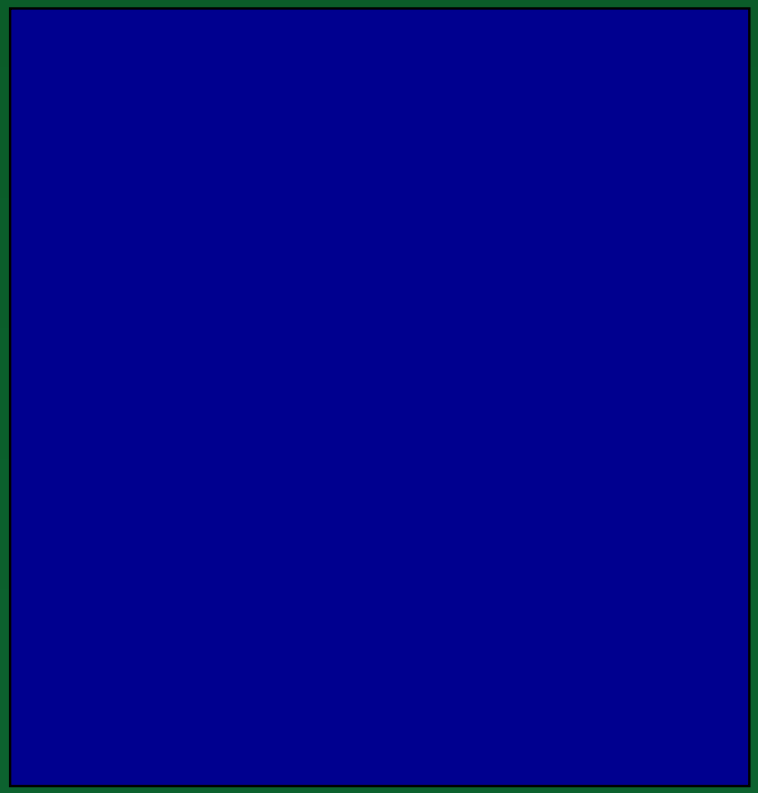
- LEMMA: $\cos\theta_i$ is algebraic
- COROLLARY (Lindemann): A non-zero θ_i is transcendental

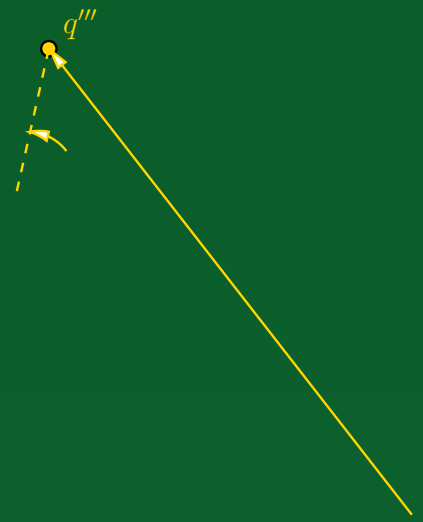
Approach for Comparing Lengths

- Assume all discs have unit radius
  - Then $d(\mu) = \alpha + \theta$, and $d(\mu') = \alpha' + \theta'$
- LEMMA: $d(\mu) = d(\mu')$ iff $\alpha = \alpha'$ and $\theta = \theta'$
- Hence, we need to ability to add arc lengths

Addition of Arc Lengths

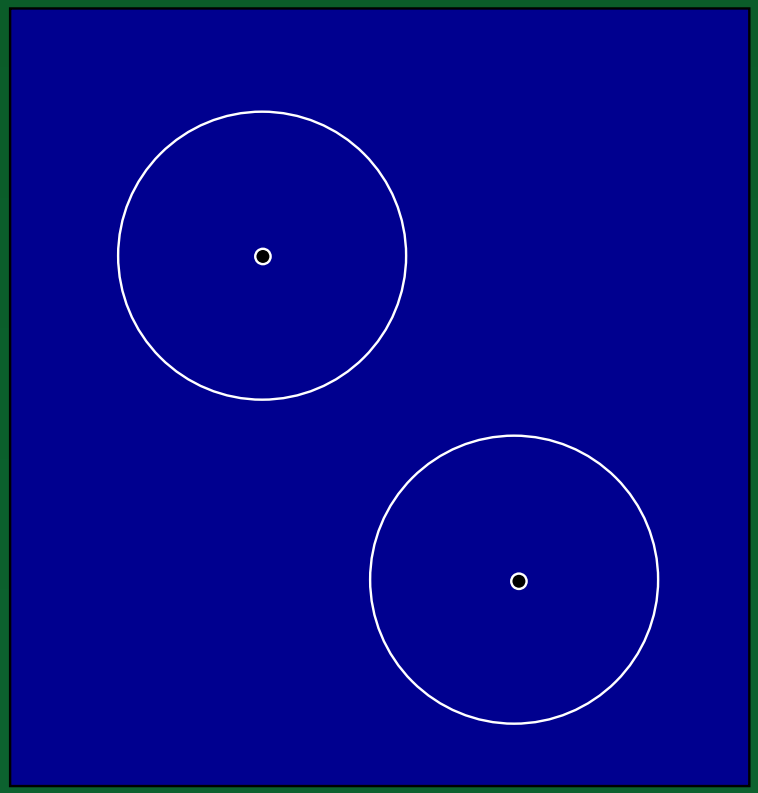
- Let $A = [C, p, q, n]$ and $A' = [C', p', q', n']$

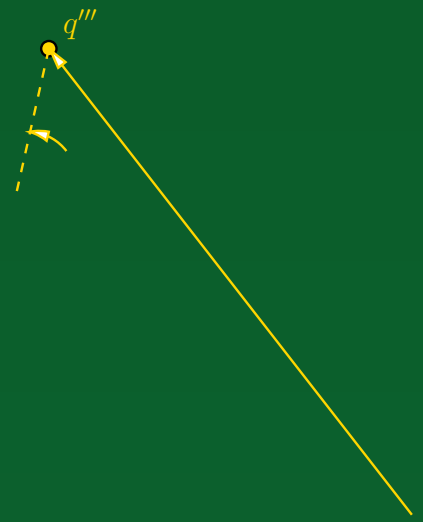




Addition of Arc Lengths

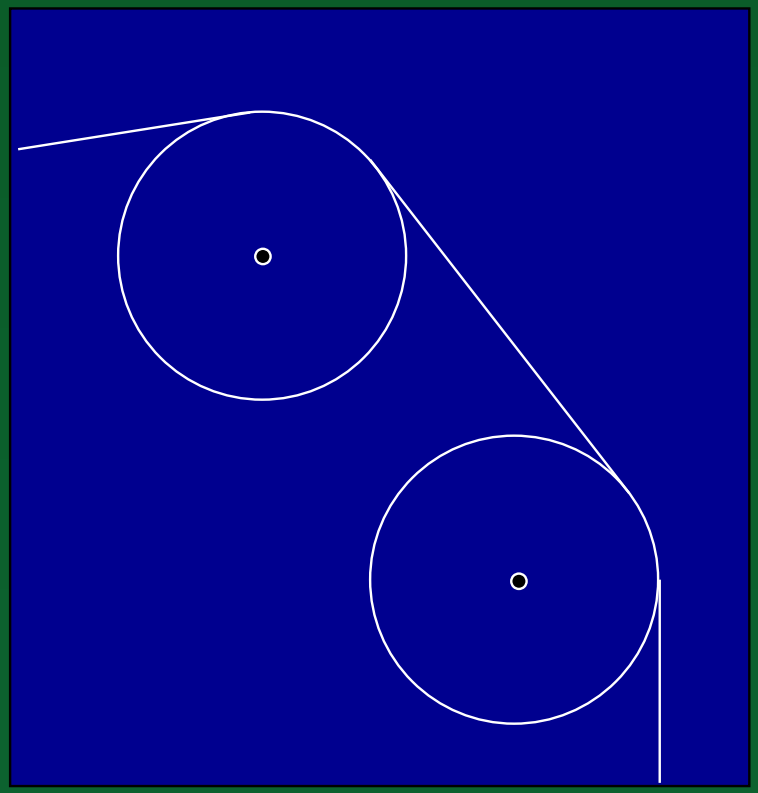
- Let $A = [C, p, q, n]$ and $A' = [C', p', q', n']$

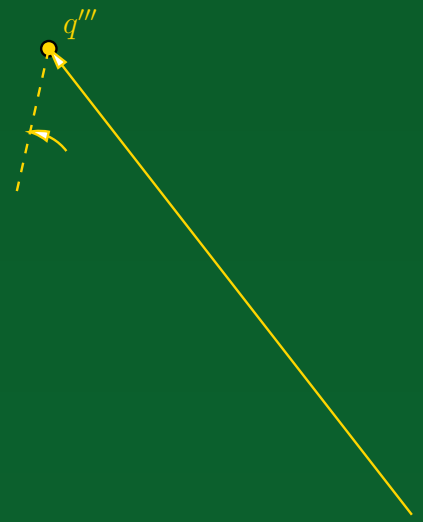




Addition of Arc Lengths

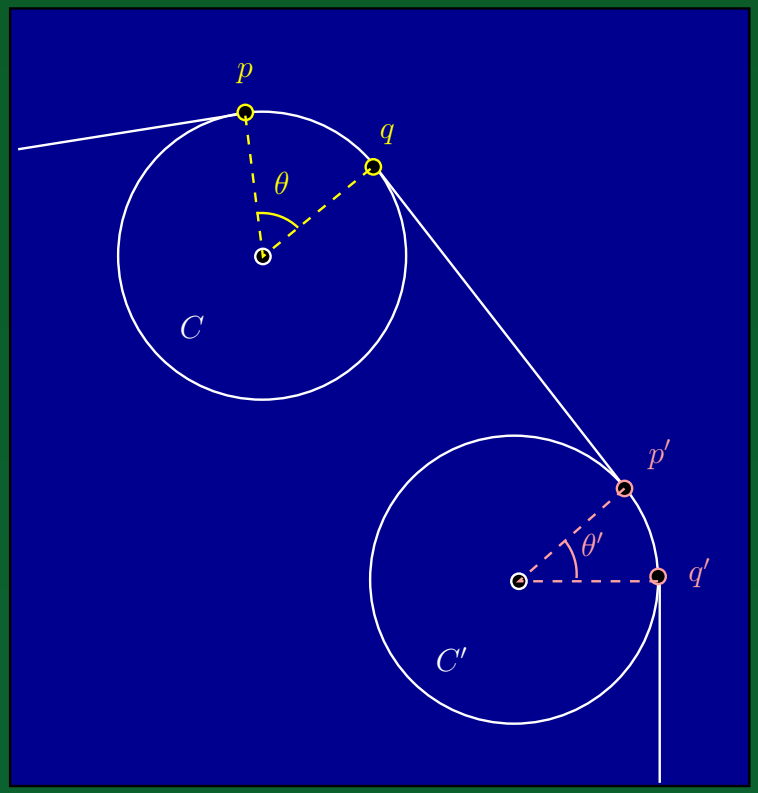
- Let $A = [C, p, q, n]$ and $A' = [C', p', q', n']$

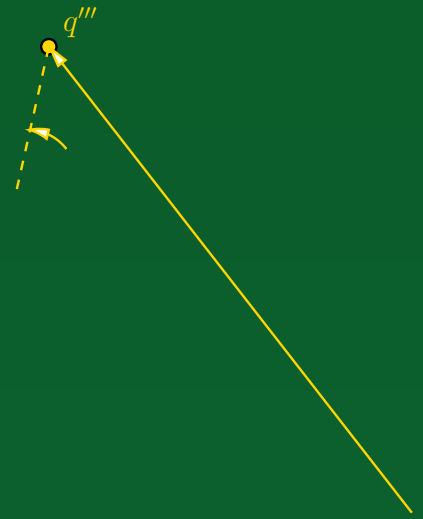




Addition of Arc Lengths

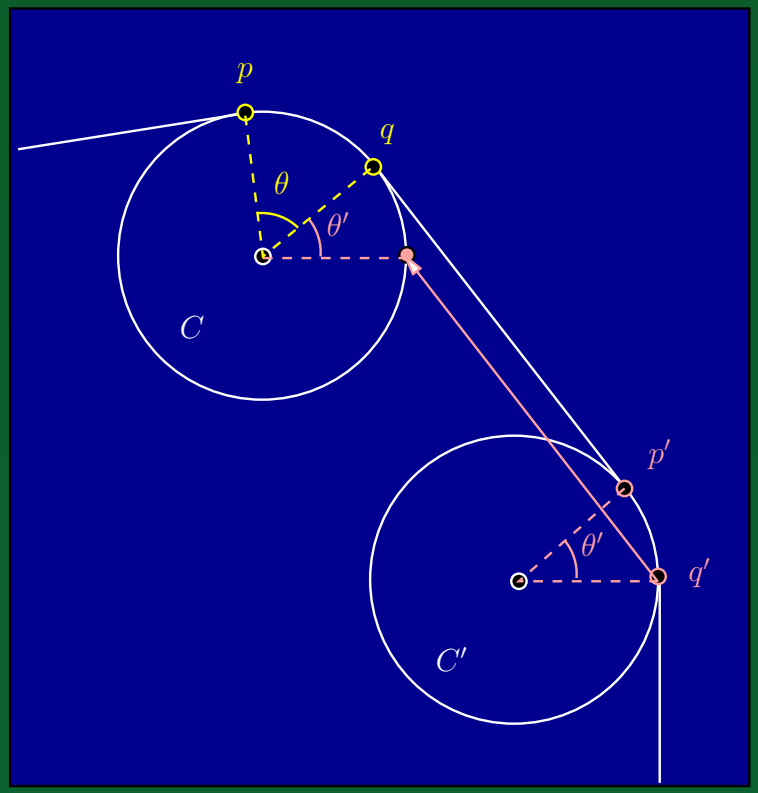
- Let $A = [C, p, q, n]$ and $A' = [C', p', q', n']$

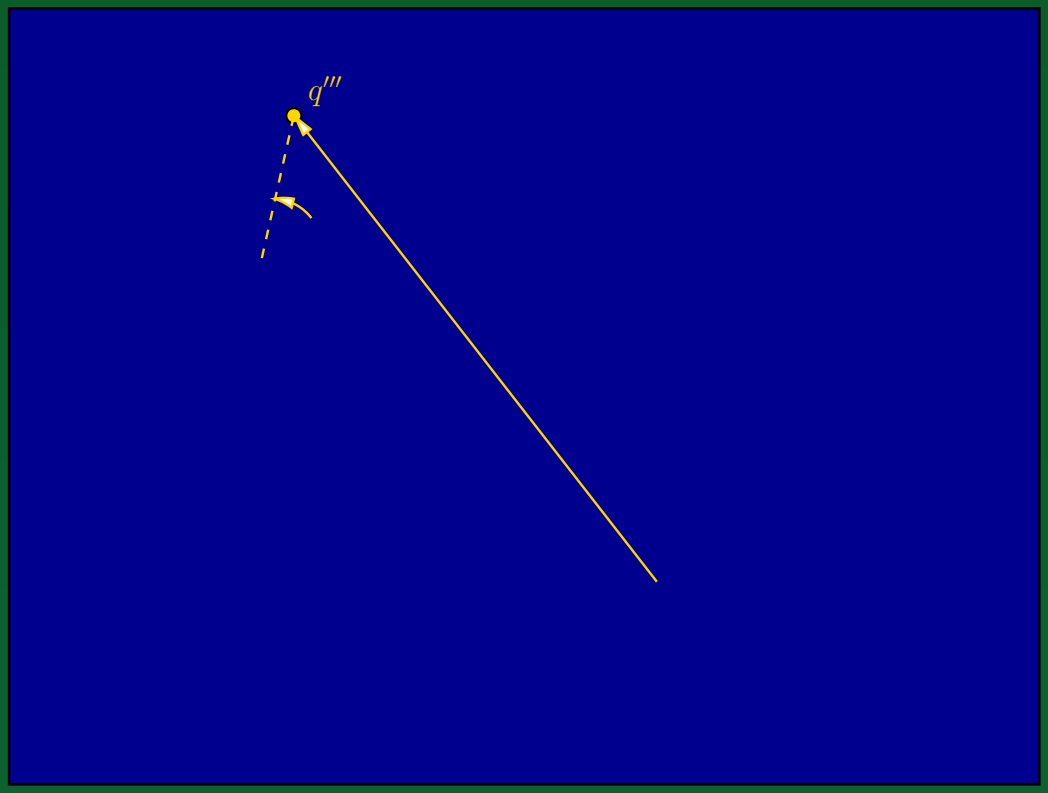




Addition of Arc Lengths

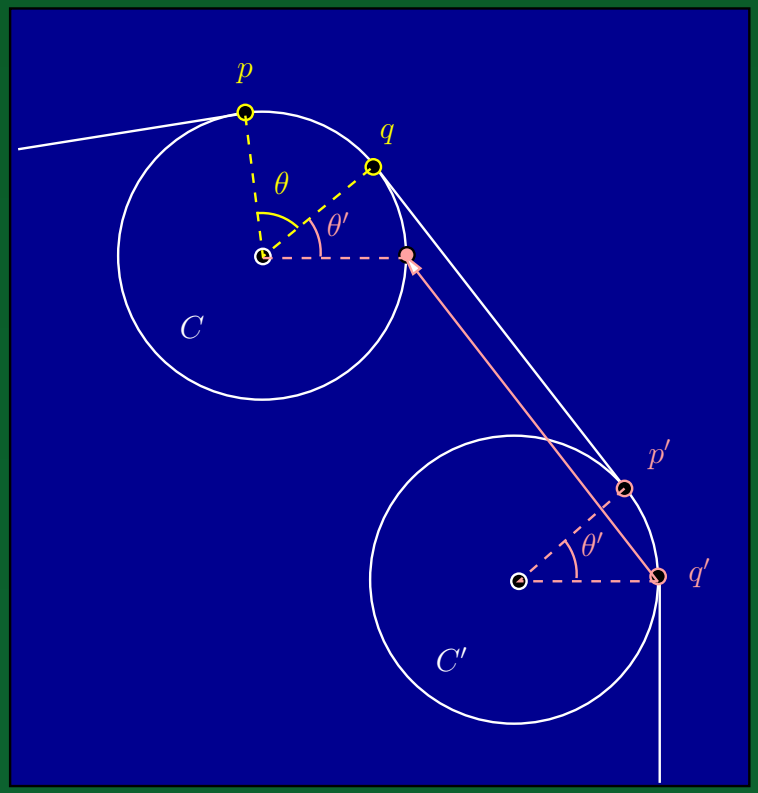
- Let $A = [C, p, q, n]$ and $A' = [C', p', q', n']$

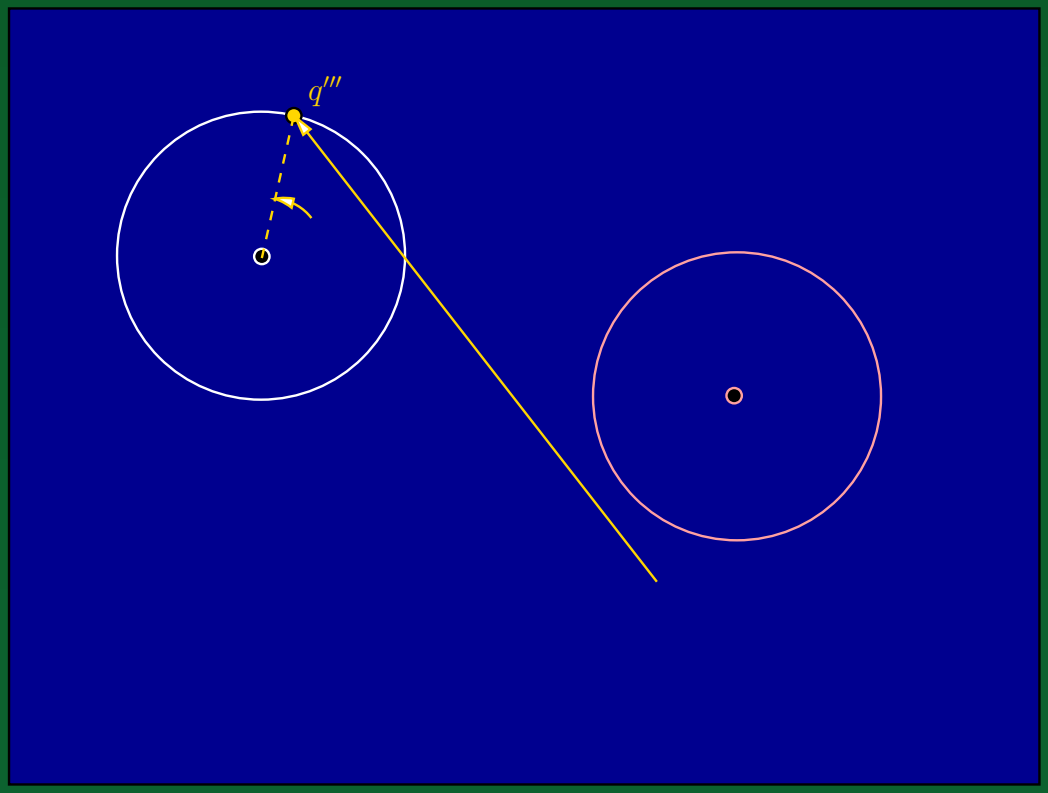




Addition of Arc Lengths

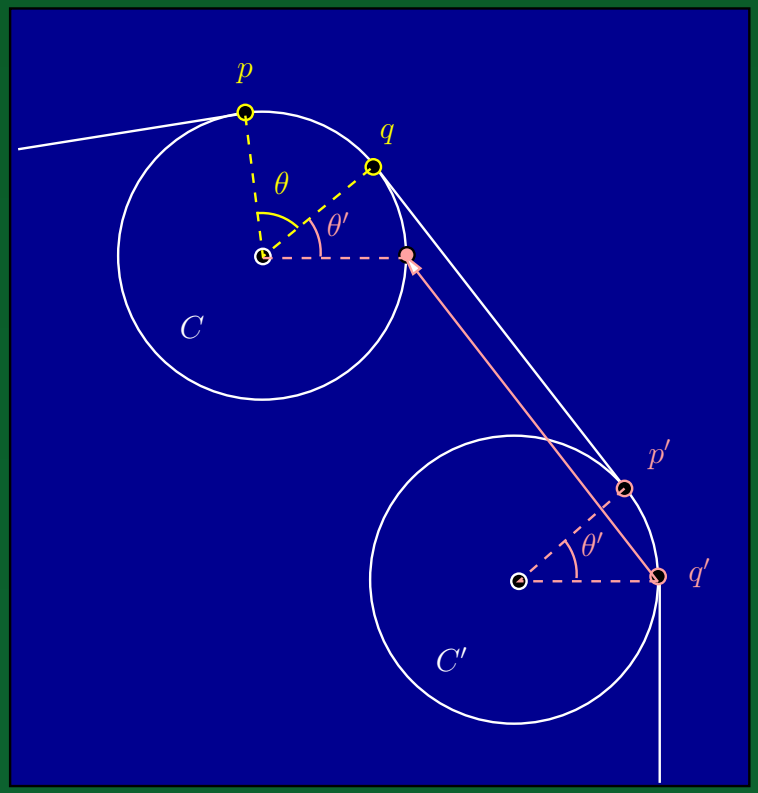
- Let $A = [C, p, q, n]$ and $A' = [C', p', q', n']$

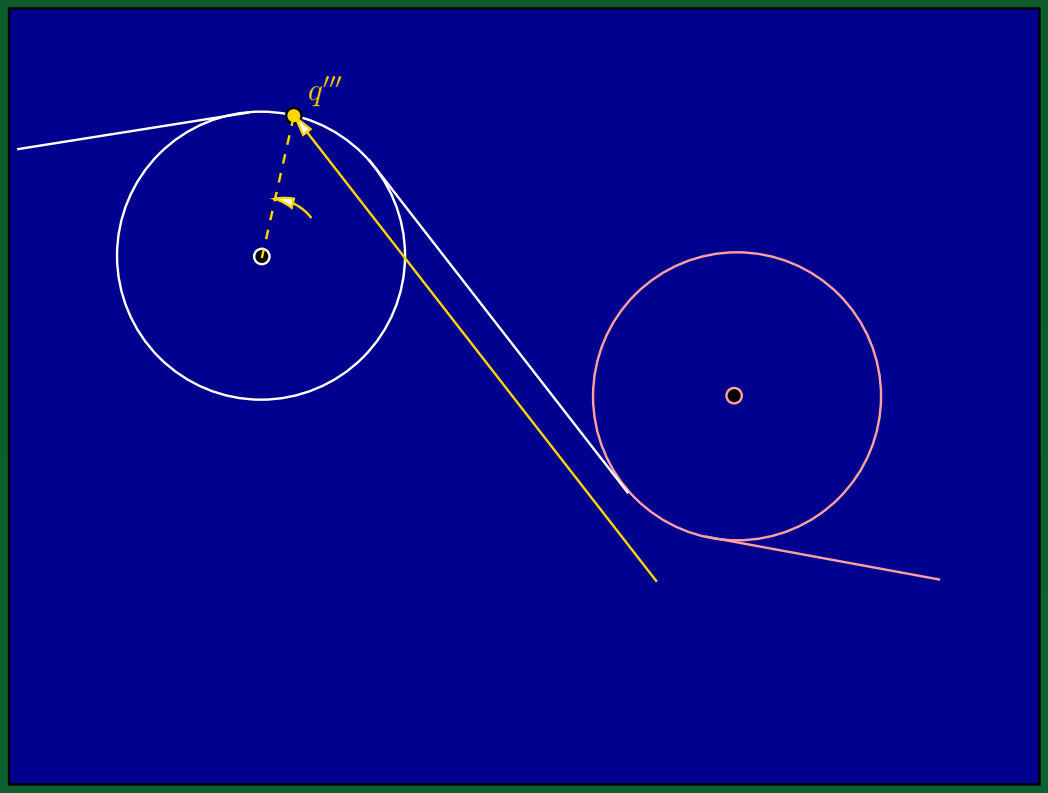




Addition of Arc Lengths

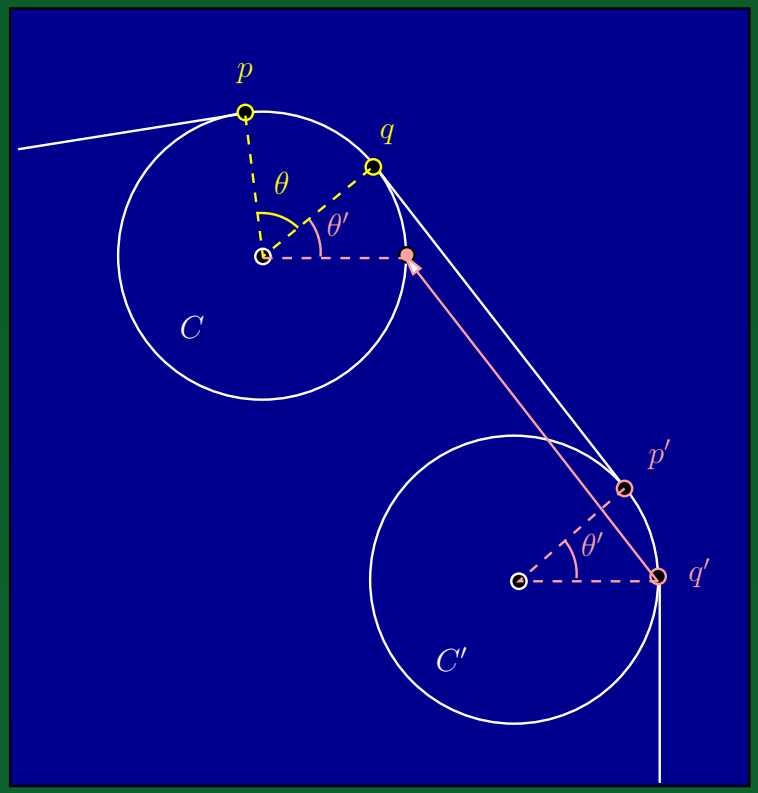
- Let $A = [C, p, q, n]$ and $A' = [C', p', q', n']$

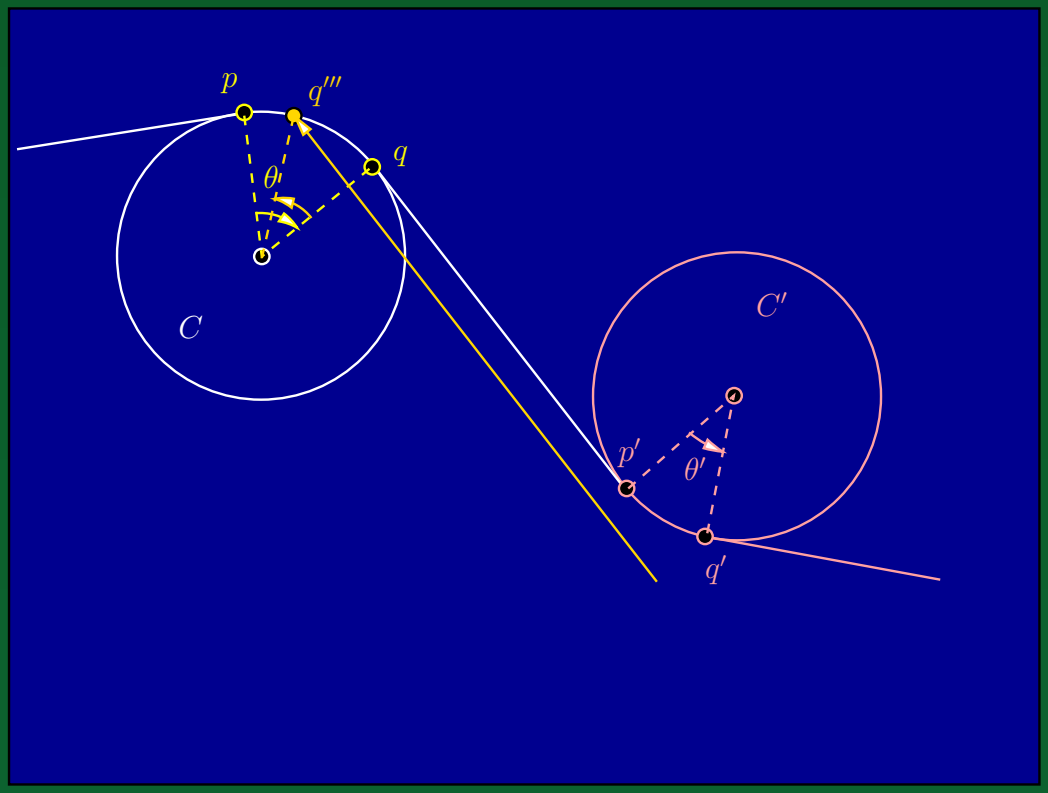




Addition of Arc Lengths

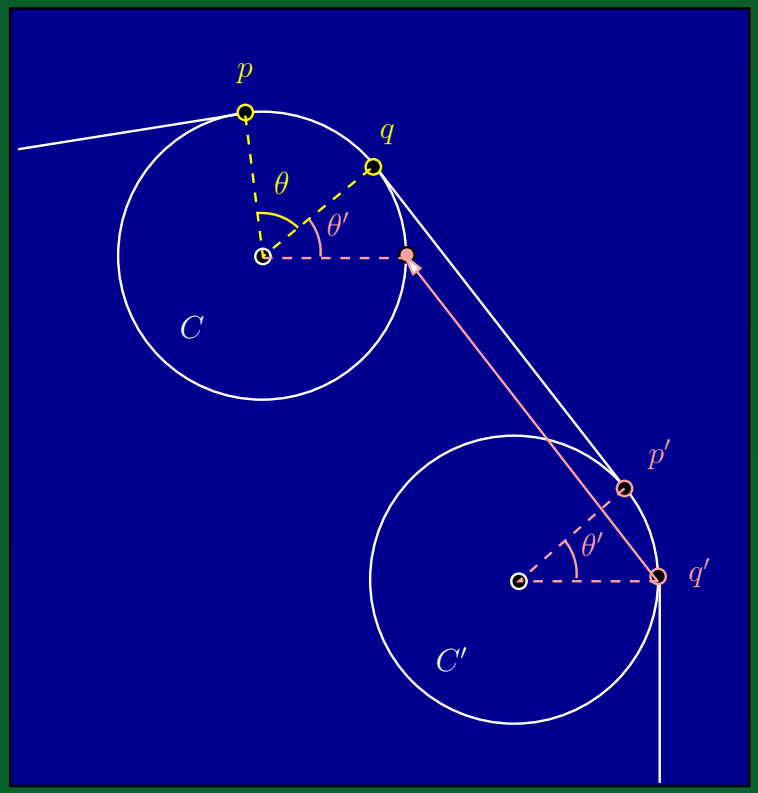
- Let $A = [C, p, q, n]$ and $A' = [C', p', q', n']$

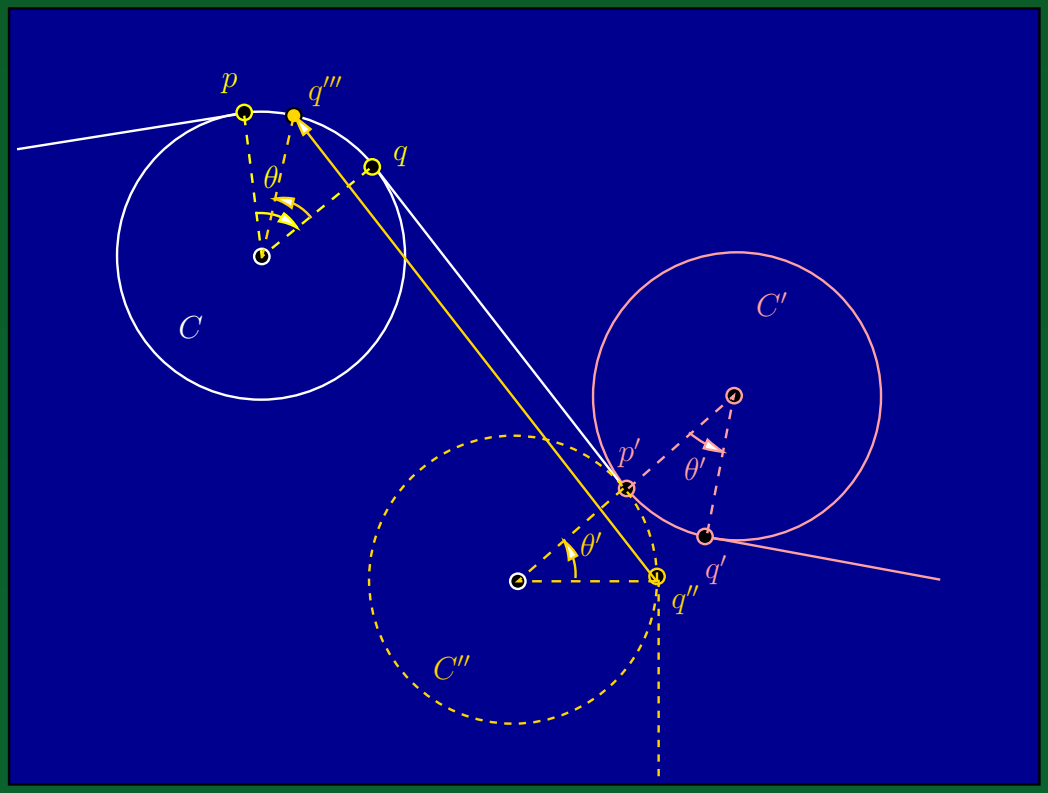




Addition of Arc Lengths

- Let $A = [C, p, q, n]$ and $A' = [C', p', q', n']$





Decidability [CCKPY'05]

- THEOREM 1
  - Shortest Path for unit disc obstacles is computable.
- THEOREM 2:
  - Shortest Path for commensurable radii discs is computable.
- No complexity Bounds!
  - Appeal to Baker's Linear Form in Logarithms:

$$\left| \alpha_0 + \sum_{i=1}^{n} \alpha_i \log \beta_i \right| > B$$

- THEOREM 3
  - Shortest Paths for algebraic discs is computable.
- THEOREM 4
  - Shortest Paths for rational discs is in single-exponential time.

- Remark: Rare positive result from Transcendental Number Theory
  - First transcendental geometric problem shown computable

Mini-summary, Part III

- Grand Challenge in EGC
  - Transcendental zero bounds
- Practical Challenge
  - More efficacious zero bounds [Pion-Yap'04]
  - Geometric zero bounds [Yap'06]

PART IV.
THEORY OF REAL APPROXIMATION

Standard Complexity Theory

- Standard Computability Theory
  - Turing machines, Recursive Functions
  - Basic foundations are widely accepted
  - Well-established complexity theory

- Wanted: Theory suitable for EGC
  - Standard theory has countable domain (finite strings, $\mathbb{N}$, $\mathbb{Z}$)
  - Need uncountable domain (Reals, Complex)
  - Higher order computability

Two Approaches to Real Computation

- Analytic Approach
  - Turing(1936), Grzegorczyk(1955), Weihrauch, Ko, etc
  - Real numbers represented by rapidly converging Cauchy sequences
  - Extend Turing machines to compute with infinite input/output sequences
  - Alternative: Russian approach (numbering instead of representation)

- Algebraic Approach
  - BSS model (Smale), Real RAMs, uniform circuit complexity
  - Real numbers directly represented as atomic objects
  - Real numbers are compared without error
  - Algebraic operators are carried out without error
  - Main interest here: $(\mathbb{R}, +, -, \times, 0, 1)$

What is Wrong?

- Known Criticisms
  - (Analytic) Only continuous functions are computable
  - (Algebraic) Unable to distinguish effective from noneffective reals
  - (Algebraic) Exponential function is not computable

- Difficulties from EGC Viewpoint
  - Zero Problem is trivial in Algebraic Approach
  - Zero Problem is undecidable in Analytic Approach
  - Wanted: theory where the Zero Problem depends on the real operators used

How We Solve Numerical Problems

- E.g., Solving a PDE model, A numerical optimization, etc
- STEP A:
  - Design an ideal Algorithm A
  - Assume certain operations such as $\pm$, $\times$, $\exp()$
  - Show that problem is solvable by Algorithm A
- STEP B:
  - Implement Algorithm A as a Numerical Program B
  - Account for numerical representation, errors, convergence, etc
  - Specify the "correctness" criteria

The New Synthesis

- Step A:
  - Algorithm A belongs to an Algebraic Model (e.g., Real RAM, BSS, etc)
  - Basis $\Omega = \{\pm, \times, \exp(), ...\}$
- Step B:
  - Program B belongs to some "Numerical Model" (Turing machines?)
  - We propose a numerical model (below)
- Critical Questions:
  - Can Algorithm A be implemented by some Program B?
  - Wanted: Transfer Theorems
- Psychological validity of Algebraic Model
  - Even numerical analysis books proceed to Step B via Step A
  - Various proposed refinements of the Algebraic model (Khoiran, Braverman, etc)
  - The algebraic and numerical models play complementary roles!

Theory of Real Approximation

- Postulate a set $\mathbb{F}$ of representable reals
  - $\mathbb{Z} \subseteq \mathbb{F} \subseteq \mathbb{R}$
  - $\mathbb{F}$ is a ring extension of $\mathbb{Z}$
  - $\mathbb{F}$ is countable and dense in $\mathbb{R}$
  - $\mathbb{F}$ has a effective representation
  - Comparisons and Ring operations are effective

- Function $f : \mathbb{R} \to \mathbb{R}$ is approximable if a Turing machine compute some $\widetilde{f} : \mathbb{F} \times \mathbb{F} \to \mathbb{F}$ such that:
  - Absolute Approximability: $|\widetilde{f}(x, p) - f(x)| \leq 2^{-p}$
  - Relative Approximability: $|\widetilde{f}(x, p) - f(x)| \leq 2^{-p}|f(x)|$

- We really want partial functions of form $f : S \subseteq \mathbb{R} \dashrightarrow \mathbb{R}$.
  - S is nominal domain
  - $\mathtt{domain}(f) = \{x \in S : f(x) =\downarrow\}$ is proper domain

Basic Properties

- LEMMA:
  - $\texttt{sign}(f(x)) = \texttt{sign}(f_{\mathcal{R}}(x,1))$
- THEOREM A: The following are equivalent:
  - (i) f is $\mathcal{R}$-approximable
  - (ii) f is $\mathcal{A}$-approximable and $Zero(f)$ is decidable
- THEOREM B: There exists f_0:
  - (i) f_0 is absolutely approximable in polynomial time
  - (ii) f_0 is not relatively approximable
- THEOREM C: [with C.O'Dunlaing] There exists g_0, h_0:
  - g_0, h_0 are $\mathcal{R}$-approximable in polynomial time
  - $g_0 \circ h_0$ is not absolutely approximable
- Composition is the basis for Zero Problems
  - So polynomial-time class must be restrained by niceness properties, e.g., Lipschitz

Transfer Theorem

- THEOREM D: The following are equivalent:
  - (I) val_Ω is relatively approximable over Ω
  - (II) For all problems F, if F is Ω-computable (algebraic model) then F is relative Ω-approximable (numerical model).

- REMARKS:
  - $\mathrm{val}_\Omega : EXPR(\Omega) \to \mathbb{R}$ is evaluation problem
  - So val_Ω is "universal" or "complete".

Connection to Analytic School

- THEOREM E:
  - Let $f : S \subseteq \mathbb{R} \dashrightarrow \mathbb{R}$. If f is computable then the following two conditions hold:
  - (i) f is partially $\mathcal{A}$-approximable
  - (ii) f has a recursively enumerable modulus cover
  - CONVERSELY, if S is regular and then (i)+(ii) implies f is computable

- By focusing on approximability
  - We are giving up precisely one thing: continuity of f

Mini-summary, Part IV

- EGC needs a Theory of Real Computation that properly address the Zero Problem

- The Analytic and Algebraic Theories can be reconciled in a new synthesis
  - Need transfer theorems
  - Focus of approximation

Summary of Talk

- We have shown
  - Why Nonrobustness must be solved
  - How it can be solved in one strict interpretation (EGC)
  - Indicated a theory of real computation that is more realistic
  - Left many open questions to be solved

Thanks for Listening!

"A rapacious monster lurks within every computer,
and it dines exclusively on accurate digits."
– B.D. McCullough (2000)

- Core Library and Papers can be download from
  http://cs.nyu.edu/exact/
- Part 3 is based on: "Shortest Paths for Disc Obstacles is Computable"
  - E.Chang, S.Choi, D.Kwon, H.Park, C.Yap. 21st SoCG, 2005.
- Part 4 is based on: "Theory of Real Computation according to EGC"
  - C.Yap. To appear.

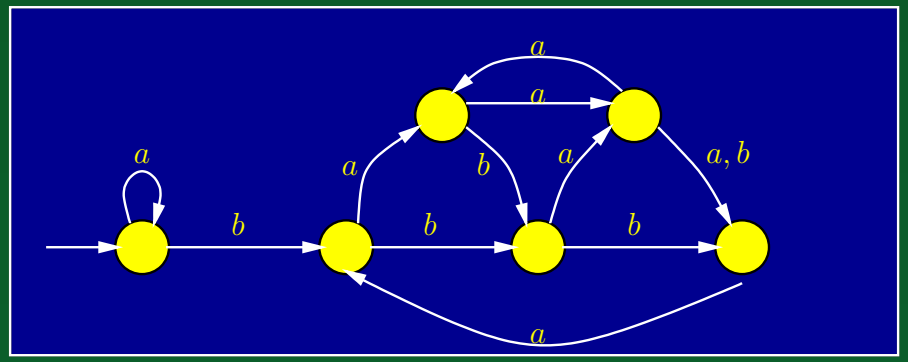
END OF TALK

Schönhage's Pointer Model

- Δ-graph G: a labelled digraph
  - Pointer machine transforms G by re-directing edges

Type	Name	Instruction	Meaning
(i)	Node Assignment	$w \leftarrow w'$	$[w]_{G'} = [w']_G$
(ii)	Node Creation	$w \leftarrow$ **new**	$[w]_{G'}$ is new
(iii)	Node Comparison	**if** $w \equiv w'$ **goto** L	$G' = G$
(iv)	Halt and Output	**HALT**(w)	Output $G\|w$
(v)	Value Comparison	**if** $(w \circ w')$ **goto** L where $\circ \in \{=, <, \leq\}$	Compare $Val_G(w) \circ Val_G(w')$
(vi)	Value Assignment	$w := f(w_1, \ldots, w_m)$ where $f \in \Omega$ and $w, w_i \in \Delta^*$	$Val_{G'}(w) = f(Val_G(w_1), \ldots, Val_G(w_n))$

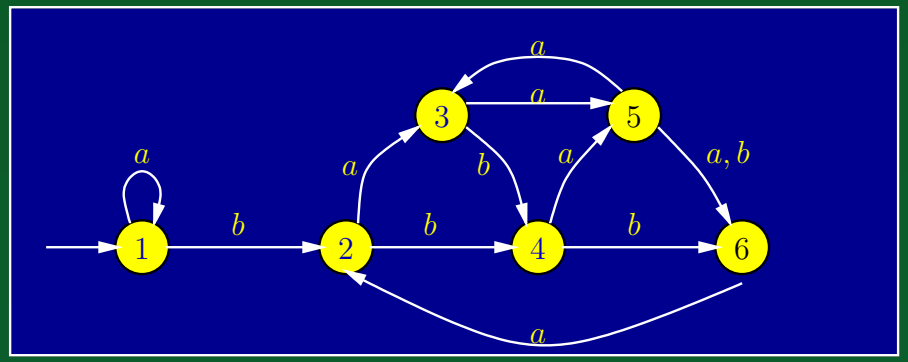
Schönhage's Pointer Model



- Δ-graph G: a labelled digraph
  - $*$ Pointer machine transforms G by re-directing edges

Type	Name	Instruction	Meaning
(i)	Node Assignment	$w \leftarrow w'$	$[w]_{G'} = [w']_G$
(ii)	Node Creation	$w \leftarrow$ **new**	$[w]_{G'}$ is new
(iii)	Node Comparison	**if** $w \equiv w'$ **goto** L	$G' = G$
(iv)	Halt and Output	**HALT**(w)	Output $G\|w$
(v)	Value Comparison	**if** $(w \circ w')$ **goto** L where $\circ \in \{=, <, \leq\}$	Compare $Val_G(w) \circ Val_G(w')$
(vi)	Value Assignment	$w := f(w_1, \ldots, w_m)$ where $f \in \Omega$ and $w, w_i \in \Delta^*$	$Val_{G'}(w) = f(Val_G(w_1), \ldots, Val_G(w_n))$

Schönhage's Pointer Model



- Δ-graph G: a labelled digraph
  - $*$ Pointer machine transforms G by re-directing edges

Type	Name	Instruction	Meaning
(i)	Node Assignment	$w \leftarrow w'$	$[w]_{G'} = [w']_G$
(ii)	Node Creation	$w \leftarrow$ **new**	$[w]_{G'}$ is new
(iii)	Node Comparison	**if** $w \equiv w'$ **goto** L	$G' = G$
(iv)	Halt and Output	**HALT**(w)	Output $G\|w$
(v)	Value Comparison	**if** $(w \circ w')$ **goto** L where $\circ \in \{=, <, \leq\}$	Compare $Val_G(w) \circ Val_G(w')$
(vi)	Value Assignment	$w := f(w_1, \ldots, w_m)$ where $f \in \Omega$ and $w, w_i \in \Delta^*$	$Val_{G'}(w) = f(Val_G(w_1), \ldots, Val_G(w_n))$

What is a Semi-Numerical Problem?

- Knuth's term
  - Geometry=Discrete Relation +Numerical Data
- Schönhage's Pointer Machines
  - Easily simulate Turing Machines
  - Naturally encodes combinatorial and geometric structures
- Algebraic Pointer Machines (over basis Ω)
  - $\mathcal{G}(\mathbb{R})$ is the set of real Δ-graphs, where each node stores a real value
  - Real Pointer Machines: manipulates real Δ-graphs relative to a set Ω of real operators
  - Instruction set: comparison of values, operations in Ω
- Computable Semi-Numerical Problem:
  - A function $F : \mathcal{G}(\mathbb{R}) \to \mathcal{G}(\mathbb{R})$
  - F is Ω-computable if it is computed by some Algebraic Pointer Machine over basis Ω

Numerical Model of Computation

- Numerical Pointer Machines
  - Numerical Δ-graphs: $\mathcal{G}(\mathbb{F})$ is similarly defined
  - Numerical Pointer Machines: manipulates $G \in \mathcal{G}(\mathbb{F})$
  - Instruction set: comparison of values, operations in $\Omega_{\mathcal{R}}$

- Approximation of Real Δ-graphs by Numerical Δ-graphs
  - $\widetilde{G} \in \mathcal{G}(\mathbb{F})$ is a p-bit relative approximation of $G \in \mathcal{G}(\mathbb{R})$ if
  - $\widetilde{G}$ and G are isomorphic
  - Each value in $\widetilde{G}$ is a p-bit relative approximation of the corresponding value in G

- Approximation of Semi-Numerical Problem $F : \mathcal{G}(\mathbb{R}) \rightarrow \mathcal{G}(\mathbb{R})$
  - There is numerical pointer machine (over Ω) that computes $F_{\mathcal{R}} : \mathcal{G}(\mathbb{F}) \times \mathbb{F} \rightarrow \mathcal{G}(\mathbb{F})$
  - Such that $F_{\mathcal{R}}(G, p)$ is p-bit relative approximation of $F(G)$