「フカシギの数え方」から広がる世界
～離散構造処理の現在と今後の展望～

湊 真一
北海道大学 大学院 情報科学研究科 教授

ご紹介ありがとうございます。北海道大学の湊です。今日はお招きいただきましてどうもありがとうございます。NTTの研究所に以前いた身として、ここで講演させていただけることを大変嬉しく思っております。自己紹介をいたしますと、私は石川県の出身で、京大の修士課程を経て厚木と横須賀のNTT研究所に大体７年ずつ、計14年間おりまして、途中アメリカに１年間行かせてもらいました。縁あって2004年に北海道大学情報科学研究科の助教授に就きまして、2010年からは教授を務めております。早いもので、北大に移ってからもう11年目になります。

北大の職員として一応の宣伝をさせていただきますと、北大は札幌駅の近くにキャンパスがあります。大都市の中心駅から徒歩5分圏内に原生林や農場がある場所というのは、皇居と北大しかないという、非常に良い環境です。四季の景色も素晴らしくて、食べ物も非常に美味しいので、機会がありましたらぜひ来ていただければと思います。

今日の講演の前半「離散構造とは」では、基本的なお話をして、後半は「フカシギの数え方」にまつわる、やや気楽なお話をさせていただきます。まず、前半のお話をする前に、私は今「ERATO」というプロジェクトをやっていて、これは科学技術振興機構（JST）の大型プロジェクト（戦略的創造研究推進事業）「さきがけ･CREST･ERATO」のなかでももっとも歴史のあるものです。この事業は、新しい科学技術の源流をつくるような研究を支援する目的で、情報系のみならず医学や物理など、全ての分野を対象とした年間4、5件のプロジェクトを扱っています。そのなかで情報系は2、3年に１件くらいしか対象にさせてもらえません。研究費は５年間の総額で10億円に上る非常に大規模なもので、メンバーも「ポスドク」を10人から15人雇っていますが、私のやっているアルゴリズム系の研究というのは、お金を積めばそれに比例して結果が出るというわけでもないものですから、いかに良い人材を集めて切磋琢磨し、相乗効果を出すかということに非常に苦心しました。既に今年が最終年度ですので、現在プロジェクトの最終評価をしているところです。

さて、まず離散構造についてお話しします。離散構造（Discrete Structure）とは、離散数学や計算幾何学の基礎をなす数学的な構造、たとえば集合や論理や証明ですとか、グラフ理論のグラフ、組合せ、順列、確率などを含めたものの総称として、離散構造と呼んでおります。およそ計算機で解けるようなありとあらゆる問題というのは、単純な基本演算を要素とする離散構造の処理に帰着されます。計算機そのものが、そういった単純な演算の繰り返しでしかないわけです。ただ、最終的には膨大な場合分け処理を要することが多いですし、これを高速に処理するということは、ありとあらゆる

応用分野に共通する基盤技術なので、現在の情報化社会に対する波及効果はきわめて大きいのではないでしょうか。

離散構造というのは数学的な概念構造です。定理を見つけて証明すれば、それは永久に不滅です、という世界なので、そういった基礎的な理論も大事なのですが、われわれがやっているのは離散構造そのものではなくて、離散構造を使って動くものを提供する、情報処理系のプロジェクトになります。処理性能を10倍、100倍、それ以上に向上させ、社会のありとあらゆる応用に繋げていくことが、われわれのプロジェクトの構想になっています。

離散構造のなかでもっとも基本的なモデルのひとつとして、論理関数というものがあります。「0,1」の変数を入力して、答えが「0,1」で出てくる、「yes, no」で「yes, no」を返答する、というものを論理関数と言います。たとえば、「a,b,c」という3つの「0,1」の入力があったとして、これをaが0と1で場合分けして、bを0と1で場合分けして、cも同じく0と1で場合分けする、といった具合に、すべてを場合分けすると、8通りの「yes, no」の答えが出てきます。週刊誌の占いのように、今日のあなたの朝ごはんはパンだったか、ご飯だったか、好きな色はなにかといったことを選択させて、最終的に吉や凶などの結果が出てくるようなものだと思っていただければよいと思います。今の例では「a,b,c」と3通りしかなかったので、2の3乗で8通りの答えしかないわけですが、変数の入力が10個になると2の10乗で1024通りになってしまう。20個あると100万通り、30個あると10億通りになってしまい、いわゆる指数爆発を起こしてしまいます。

ところが、人間が考えることというのはそこまで複雑ではなくて、規則性があったり、偏りがあったりします。なので、そういった規則性や偏りを上手く使って圧縮してやると、コンパクトなかたちで表わすことができます。これを「BDD」（Binary Decision Diagrams）と呼んでいます。場合によっては、数10倍、数100倍、それ以上の圧縮が得られる例もありますが、最悪の場合、まったく圧縮できないこともあります。BDDは、1950年位からずっとありましたが、1986年にカーネギーメロン大学（CMU）のBryantという先生が画期的な演算アルゴリズムを提案して以降、急速に広く使われるようになりました。この演算アルゴリズムは、長期間、情報科学の全分野での最多引用文献となっていて、現在でも、文献登録サイト「CiteSeer」で世界9位という引用件数を誇っています。通常、圧縮データを使って検索したりいろいろな処理をしたりするときには、解凍しなければならないのですが、これはそうではなくて、圧縮した状態の2つのBDDを与えて、その間の論理演算をして、圧縮したままのBDDを直接返すというものです。つまり、一度も解凍しないそのままの圧縮データ量にほぼ比例する時間で計算できるという点で非常に画期的でした。

私はNTTの社員だったころ、LSI研究所というところにいました。そこで大規模なLSIの設計をし、データを圧縮しながら計算するということをやっていたときに、これは非常に重要な技術でした。1990年代には、LSIの設計の分野でこのデータ構造の技術が非常に発展しまして、ポピュラーな技術

となったのですが、人工知能やデータベースなど他の分野ではあまり使われていませんでした。ところがこのBDDの技術は、2000年代以降にコンピュータの主記憶が大規模化して安くなり、今まで入りきらなかったような大規模データがメモリに乗るようになって、そのメモリの上で圧縮したまま計算するというアルゴリズム技術が非常に脚光を浴びてきたことから、他の分野でも使われるようになっていきました。
これは、実際のBDDの例ですが、同じ論理関数であっても、圧縮できる場合とそうではない場合では結果が大きく違うんですね。

次に「ZDD」についてお話ししますが、その前にある論理関数について考えたいと思います。「真理値表」と呼ばれているものですが、「a,b,c」の3つのアイテムがあって、aかつbかつ¬（ノット）cまたは ¬bかつc、という論理関数があったとします。aとbとcなので、「0,0,0」から「1,1,1」まで8通りの入力の組合せがあって、それに対するさまざまな答えが出てきます、ということを表わしたものです。たとえば、スーパーマーケットを想像していただきましょう。1人目のお客さんがaとbという商品を買いました。2人目のお客さんがaとcを買いました。3人目のお客さんはcだけを買いました。今日一日でこの3通りのお客さんがありました。「トランザクション･データベース」と呼ばれるものですけれど、スーパーマーケットの購入履歴を想定した組合せ集合です。真理値表はこういったことを表わしています。つまり、組合せの集合と論理関数には、1対1の対応関係があるんですね。つまり、論理関数を使って、組合せ集合を表わすことができるわけです。さらに面白いことに、組合せ集合における集合演算、「Union」や「Intersection」や「補集合」といったものが、普通の論理関数「or」や「and」や「not」の論理演算と対応関係にある。つまり、普通の論理関数＝BDDを効率良く処理できるのなら、それを使って組合せ集合も集合演算ができるのです。これによって、今まで使われていなかったさまざまなデータベースの分野などに、BDDが使われるようになっていきました。

ZDDは、今お話しした組合せ集合を、より効率よく表現するために出てきた、BDDの派生系のようなものです。ZDDは、私が厚木の研究所にいた93年ごろに発明、考案して命名しました。ZDDがどういったものかというと、いままでのBDDの場合には簡約化があり、xが0の時と1の時で答えが同じだったら、xがあってもなくても答えは同じで無駄なので、削除して直結するのですが、ZDDの場合では、xが1の時にもし空集合に繋がっていたら、その場合を削除して0側に直結するという、ZDD特有の左右非対称な簡約化規則を使っています。一見不自然な感じがするかもしれませんが、先ほどのスーパーマーケットの例で考えていただくと、xは牛乳のような商品になります。一方が牛乳を買った人の集合で、他方が買わなかった人の集合です。買った場合が空集合に繋がっているということは、今日牛乳を買った人がいなかったということです。つまり、牛乳がスーパーにあったとしてもなかったとしても、結果は変わらないので、ZDDの簡約化では、牛乳そのものがスーパーになかったことにして、それ以外の商品について考えます。

BDDでは、この例のような場合を、今日牛乳を買った人も買わなかった人も、牛乳と一緒に会計をした商品は全く同じだったと考えます。しかし、牛乳以外の商品が全く同じだったということはあまり起こらないので、ZDDでは、よく起こらない事例を圧縮するよりは、よく起こる事例に目を向けて圧縮するのが最良であると考えます。つまり、よく起こらない事例についてはそのまま残しておいて、よく起こる事例のみを圧縮します。スーパーマーケットには、商品が数100とか数1000置いてありますが、１人のお客さんが買う商品の数は、たかだか数10個くらいしかないわけです。全商品の1%ほどしか選ばれず、99%は選ばれない。1%しか選ばれないとすると、その逆数である100倍ほどは圧縮できる可能性があるということです。

ZDDはBDDの改良技術として、現在世界的に広く使われています。最近ではデータ･マイニングの分野に応用され、他にもLSIの設計式やグラフに関する種々の問題への応用など、画期的な有効性を示しています。一例として、データ･マイニングへの応用についてお話しいたします。先ほどのスーパーマーケットのデータのようなものだと思ってください。データ･マイニングでもっとも基本的な「頻出アイテム集合」という問題があります。これは、今日は全部で11人のお客さんが順番に来て、それぞれアイテムを買いました、このなかで頻出アイテム集合を抽出する、つまり、アイテムの組合せのなかでたくさん現われる組合せはどれですか、という問題です。たとえば、7回以上買われたものの組合せを全部出してください、とパラーメーターを入力すると、「ab」という組合せが7回以上出てきます。「bc」も「aだけbだけcだけ」という組合せも7回以上出てきます。それらを、すべて列挙することが目的となります。この使いかたとして、最初にものすごく大きい頻度を入れてやると、自明な結果しか出てこないのですが、だんだんその頻度（閾値）を下げていくと、たくさんヒットするようになり、最後はほとんど指数的な組合せの数だけ出てきます。なので、実際には100位の組合せが出たところで止めて、面白い組合せを探します。たとえば、ビールとおつまみが一緒に出ているのであれば、それらをセットにするとか、ビールと紙おむつが一緒になっているのは面白いよね、といった案配で探すのがデータ･マイニングです。

計算結果をメモリ上で圧縮してZDDで表現し、それを返すというLCM-ZDD法と呼ばれているものもあって、これは、私と、世界最速のアルゴリズムLCMの開発者である国立情報学研究所（NII）の宇野先生の2人で共同開発しました。たとえば、先ほどのデータには、頻度が7以上の組合せがたくさんあるのですが、ZDDを使わない手法ですと、これらを全部ファイルに書き出していくのですが、LCM-ZDD法ではこれを先ほどのZDDをメモリ上に構築して返します。これは、ルートから1を表わす終端までのそれぞれの経路が解を表わしている。aがあってbがあるのは、「ab」を表わしている。aがあってbがないのは「a」だけを。aがなくてbとcがあるのは「bc」を。同じく「b」だけ「c」だけがあって、この5通りのパスを表わしています。つまり、この解のデータをZDDのパスの情報として圧縮して表わして、それをメモリ上に置き、ポインタだけを返すという作業です。実際にやった例ですが、閾値をだんだん下げていくと、たくさんヒットするようになって、解の数が指数爆発的に増えていきます。従来の方法ですと、これをただ出力していくだけなので、大きくなると時間がかかってしまうのです

が、ZDDを使うと圧縮されているのでそれほど増えない。計算時間もそれほど増えないので、従来とは100倍ほどの差が出てきます。ZDDをこのように使用するととても効果的です。

しかしながら、たとえ答えのパターンが15億通り出たとしても、人間はそんなにパターンを見ることはできないでしょう、と疑問に思うかたがいらっしゃると思います。ですが、ZDDは巨大なものを圧縮するだけでなく、圧縮したものどうしを演算することができます。なので、昨日と今日のデータがあるとして、昨日のデータで頻出なものを10億通り出し、今日も同じように10億通り出したとすると、それぞれの共通部分や差分をとった時に、例えば100通りといった非常に限られた数の特徴的なパターン集合が得られます。従来の明示的な表現方法を使っていると、非常に頻度が高い限られたパターンだけしか調べられなかったのですが、ZDDを使うことで、それ以上の細かいことがわかるようになりました。

そのZDDにまつわる話題なのですが、Knuth先生の世界的な名著に私のやっていたZDDが取り上げられました。あるとき突然メールが来て、載せるためのチェックを頼まれて、びっくりして2カ月ほど仕事を止めてそのことばかりしていた覚えがあります。余談ですけれど、Knuth先生から届いた小切手があります。これは2ドル88セントという金額的には全然意味のないお金ですが、Knuth先生は、先生の教科書の間違いを発見／報告して、それが今まで指摘されていなかった間違いなら、誤り指摘一件あたり2.56ドル、つまり2の8乗セント差し上げます、というシステムを作りました。同じように、改善提案が採用されたときは、2の5乗セントで0.32ドル貰えます。その金額と発見者の名前、指摘されたページと章、そして日付と通し番号が書いてある小切手を送ってくれるんですね。この話はWikipediaのKnuth先生の項目にも書いてあると思いますが、私の場合は誤り指摘一件と、改善提案一件で、小切手を頂きました。

ところが、2008年10月までは、パロアルトの近くに実在する銀行の本物の小切手だったんですが、その後のリーマンショックで銀行が潰れるかもしれないとなったときに、Knuth先生も、潰れてしまったら意味がなくなるし、そもそも誰も換金しようとしないので、本物の小切手を送ることはやめて、以降は架空の惑星にある架空の銀行の預金証明書になりました。16進法が使われている架空の惑星から送られてくるそれには、0xドルと書いてあって、つまり1.00ドル(16進ドル)が預金されましたよ、という意味を表わしています。ちなみに今、ERATOプロジェクトに関係している人のあいだで、Knuth先生の教科書の輪講をしていて、いろいろ間違いを見つけたものを報告しまして、確か7人が小切手を貰っています。2010年には先生のお宅を訪問して、いろいろとアドバイスをいただいたりもしました。

技術的な話に戻りますが、ここにZDDの演算リスト（演算体系のようなもの）があります。これには、「Union」「Intersection」や「差集合」の演算があります。また、それぞれの集合に要素を加える／加えないという演算や、要素の数を数えるといのもあります。演算リストに黄色で書いた部分

がありますが、この部分は、論理関数の代数＝algebra(アルジェブラ)と1対1の関係があるため、対応づけられるのですが､それ以外にも新しい演算があり､PとＱの集合の直積をとるといったような、組合せ集合独特の演算も考えられます。ほかには、その逆演算である割り算もあります。割り算をした商と余りが書いてあります。これは、論理演算には直接対応づけられない組合せ集合特有の演算です。これらの演算は、非常に効果的なさまざまなアプリケーションがあります。Knuth先生はこの辺りの話にも大変興味を示されて、実際に教科書にも多くのページを割いて書いていらっしゃいます。もともと私が93年か94年頃にこれについて研究していたときは、「unate cube set algebra」と呼んでいたのですが、Knuth先生は、その名前はイケていないので、もう少しイケてる名前にしようと提案されて、「family of set」（集合族）に関する演算だから「Family Algebra」と呼ばれることになりました。ついでに言うと、「ZDD」という名称も元々は「Zero-suppressed BDD」だったので「ZBDD」と呼んでいたのですが、Knuth先生は、大事なものの名前の文字数は短いほうがよいと言うので、「ZBDD」を「ZDD」に改めました。

プロジェクトの話に戻りますが、われわれのプロジェクトでは、BDD／ZDD技術を新しい切り口として、さまざまな離散構造を統合的に演算処理するような技法を体系化して、分野横断的かつ大規模な実問題を高速に処理するための技術基盤を構築したいと思っています。底部が「Computer science」などの概念的な「理論」の部分だとすれば、上部は個別の工学的応用に特化した「技術」領域になります。われわれが目指すところは、そのどちらでもない中間の（実装技術であるところの）「離散構造処理系」の部分です。簡単に言うと「Ａｒｔ」ですね。これは芸術という意味ではなくて、Knuth先生の教科書が『The Art of Computer Programing』であるように、技法（state-of-the-art）という意味での「Ａｒｔ」です。つまり、概念的な「理論」だけでなく、動くものを作り、実装した上で提供するということです。ただし、この「Ａｒｔ」層においては、アドホックに作るのではなくて、技術基盤としての完結さや汎用性や美しさを重視しています。

本当にあるのかどうかはわからないのですが、「Science」と「Engineering」のあいだに「Ａｒｔ」層がある、と私は主張しました。「情報科学」は「Science」じゃないだろうと言われることがよくあります。また、お互いの分野間でお金の取りあいや攻撃をしあうといったことがあるために、「Science」であることに疑いをもって言ってくる人もいます。しかし私が申請書で、「Science」と「Engineering」のあいだに「Ａｒｔ」というものがありますと言ったところ、それは良い考えだ、ということに落ち着いたようです。私が思うに「Science」とは、神がいたかどうかわかりませんけれど、大自然の摂理を解明して真理を探求するものでして、それに対して「Ａｒｔ」は人間がつくった美しいものを体系化していくことなので、本当にそうなのかはわりませんが、そういった区分があると言い張ることにしています。

今あるZDD自体は20年前につくられたものですが、時代の変化に応じていろいろなものに使われるようになっていきました。しかし、まだまだ使われる余地が多く残されているので、それを使えるよ

うにすることが短期的な技術上のポイントです。ただ長期的には、ワード単位のデータやベクトルなどのより高度なモデル、文字列の集合、順列の集合、分割の集合やツリーの集合やネットワークの集合などのモデルに使っていく見通しをもっています。

それらの高度なモデルに対しても、さらに高度なZDD風のデータ構造やそれに関する演算体系があるはずなので、長期的な計画では、それらをみつけて構築することによって、より豊かな応用に繋げていこうという試みがあります。こうした研究はずっと続いていくだろうとも言われています。たとえば、文字列の集合を表わすZDD風の構造として「Sequence BDD」というものがあります。これは組合せではなくて、文字列です。Aが3回繰り返すとかbが2回出てくるなどといったものすべてを区別するわけです。こうした集合をZDD風に表現して処理する技術が2009年に見つかりました。この基本演算にも、ZDDの演算と非常に似た演算体系があることがわかっています。ZDDが出てくる前に使われていた「Suffix Tree」など、文字列に関するさまざまなデータ構造がありまして、それと融合して新しいものが出てきています。順列に関するZDDもありまして、順列の処理というのは、ルービックキューブや15パズルやトランプ、あみだくじやソーティングネットワーク、さらには符号理論の応用や、量子の計算とも関連があります。これは「permutation decision diagrams」（πDD）と略して呼んでいるZDDふうのデータ構造で、順列の集合を表わしてそれを演算するといったような演算体系です。やはり、同じようなalgebraが作れます。

さて、前半のお話はこのくらいにしまして、「フカシギの数え方」についてお話ししたいと思います。「フカシギの数え方」は、お台場の日本科学未来館で行なったERATOプロジェクトの成果展示のひとつです。目的は、情報科学やアルゴリズム技術といった抽象的なものを、一般の方や青少年にわかりやすく見せることでした。こういった試みは、いままであまりやられてこなかったので非常に難しいものでしたが、なんとか理解してもらいたいと、JST本部からも打診がありまして、実行することとなりました。日本科学未来館はJSTの組織であり、全国の科学館の総本山のようなところで、新しい展示手法の開発をミッションとしているんですね。未来館には一流のクリエイターや、展示に関わる専門家がいるので、その人たちと意見をたたかわせるなかで、「インタラクティブにペンでなぞる」「身体よりも大きな表をつくって視野の広さで迫力を出す」「体重をかけると圧縮される体感型にする」「研究者の顔写真を展示することによって、生身の人間が研究している現実感をもたせる」といったさまざまな工夫をこらした展示手法のアイディアが生まれました。アニメーション動画は、数え上げがいかに大変かということを、人間の寿命と対比させてわかってもらうために工夫してつくりました。これをYouTubeで公開したところ、大ヒットして、現在では150万回以上見られています。ちなみに、無断転載なのですが、ニコニコ動画では、45万回以上見られています。でも、ニコニコ動画の方がコメントが流れるので見ていて面白いです。

「フカシギの数え方」は、おねえさんが子供に組合せの数え方を教える内容になっています。最初は簡単なn×n格子グラフの道の通りかたを教えるのですが、途中からだんだん難しくなってくるので、

スーパーコンピュータで計算を始めます。スーパーコンピュータを使ってもすごく時間がかかってしまう問題(9×9)を、2、3日で片付くだろうと予想するのですが、待っているあいだに季節が過ぎて、子供たちはどんどん大きくなってしまいます。気がついたらなんと6年も経ってしまっていました。でも、おねえさんはそのあいだに数え続けていました。と言っても、実際はコンピュータが数えているわけですが、子供たちはおねえさんの姿を見てドン引きします。その後、急にシリアスになって無駄に感動的なのですが、計算を続けると死んでしまうと知りつつも、おねえさんは10×10の計算に取りかかります。ついには、計算に25万年もかかってしまって、おねえさんはロボットになってしまいます。ところが、25年かかっていた計算が、今の技術ではたった数秒で数えられるんです。そのことをおねえさんに教えてあげたいと言って幕を閉じます。

おかげさまで、サイエンス系のコンテンツではきわめて異例の数である150万回以上の再生回数となりましたが、これは、以前ニコニコ動画で150万から200万回の再生回数に到達した「人工衛星はやぶさ」のパロディ動画に匹敵する結果です。しかもこの映像は、科学未来館とERATOプロジェクトが発注してつくったものですので、著作権はわれわれがすべてもっています。YouTubeやニコニコ動画の映像は、夜にしか再生数が伸びていないことが結構あるのですが、この映像は昼も夜も満遍なく見られています。さらに、字幕があることによって、英語圏のみならずアラビア語やロシア語やイタリア語といった、さまざまな言語のコメントがあって、反響を非常に感じることができました。

このアニメーションに出てくる問題は、同じところを2度通ることはできない、ということで「self-avoiding walk」と呼ばれています。最短経路の数え上げであれば、高校の数学レベルとして大学の入試にも出てくる簡単な問題なのですが、最短でない経路を含めると突然難しくなって、現時点で計算式や漸化式は見つかっていません。アニメでは、25万年かけて出した結果が1の桁まで正確に表示されているので、きっと公式があるのだろうと思ったかたもいらっしゃるでしょうが、残念ながら公式は見つかっておらず、効率よく数え上げるしかない問題なのです。

先ほどのKnuth先生の教科書にも演習問題として載っていて、8×8の場合ZDDを使うとたった3万ノードくらいでできますよ、といったことが書かれています。このアルゴリズムでは、辺に変数番号をつけて、辺を使うなら1、使わないなら0と場合分けしていきます。1番の辺を使う／使わない、2番を使う／使わない、3番を使う／使わないといった具合に場合分けしていって、ちゃんとパスになっていれば1を入れて、パスになっていなければ0を入れます。最終的には、これを圧縮してZDDを作ります。ですが、すべてを場合分けしていると指数時間がかかってしまうので、途中の経過を省略し、共通する部分があれば共有できる、というふうにして高速に求めています。例として、アメリカの州を西海岸から東海岸まですべて廻る最短ルートを求めたものや、日本地図を同じように求めたものがあります。われわれの研究プロジェクトでは、21×21の格子グラフの数え上げに成功しています。それまでは19×19が世界記録だったので、一気に2段階更新しました。この記録を数列のオンライン百科事典サイト（Integer Sequences）に登録しようとしたところ、すでに「フカシギの数え方」がリン

クされていました。この数え上げは歴史のある問題で、12×12はKnuth先生が1995年当時の世界記録として出しました。われわれが21×21の107桁の数を数え上げたのは2012年です。

われわれは今まで、格子グラフに限らず計算をしていたのですが、格子グラフに限ればもっと大きな数え上げができそうなので、もう少し継続してみることになりました。さらに正月休みに突然、「良い解き方を思いついたので見てほしい」と一般市民のプログラマーからメールが来まして、たまたまその人が札幌にいる人だったので、休日に北大に来てもらって一緒に話しをしました。すると、プロも驚くアイディアを提案されたので、アマチュア・プログラマーの肩書きで共著者に入ってもらいました。しかし、いつの間にかノルウェーの大学の人たちがn=24まで解いてしまっていました。やはり悔しかったのですが、われわれのほうが彼らより速く解けそうだということがわかったので、プロジェクトの総力を結集して25までの計算を成功させ、世界記録を奪還しました。さらに最近26まで更新することができました。その時に、主記憶2TBのPC2台を大人気なく1週間占領しましたが、そのことよりも、それまでにさまざまな工夫を凝らしていたという過程が大切なのです。この話は、情報処理学会の解説記事（2013年11月号）に掲載されていますので、興味のあるかたはお読みになってください。ちなみに、読者アンケート評価で2013年の全記事中第1位をとりました。

この時、実際につくったプログラムは「Graphillion」という名前で、オープンソフトウェアとして公開しております。〇〇phillionというのは、「million」や「billion」のように非常に大きな数のイメージなので、何十億個ものグラフを扱えるという意味を込めて命名しました。graphillion.orgで検索するとソフトウェアが出てきますので、興味のあるかたは是非試してみてください。アニメーションとGraphillionについては紹介記事にもなっていまして、「実は９カ月も経ってから続編が公開されていたことはあまり知られていない。続編は『数え上げおねえさんを救え』と題し、グラフに対して検索や最適化を高速に行なえるGraphillionなるPythonのモジュールを紹介している」と書かれています。「数え上げおねえさんを救え」は、非常にマニアックな続編なのですが、再生回数が2カ月程前に8千回だったので、現在では既に1万回を超えていると思います。

私が関わっていたわけではないのですが、今年、2014年の北大の入試に格子グラフの数え上げの問題が出ました。これは高校の数学の範囲で解けないといけないので、最短経路だけを求めさせて、遠回りは許さない問題になっています。この中で、ある特定の区間を通る経路／通らない経路は全部で何通りあるか、ということを求めさせて、次に、この区間は他よりも早く通ることができる／時間がかかる、と想定したときに、すべての可能な経路から無作為に１つの経路を選んだ時に、出発点から到達点まで何分かかるかの期待値を求めよと、という問題です。単なるパズルとしての問題ではなくて、カーナビゲーションなど実際の応用にこうした数学的な問題が繋がっているということを入試問題でも意識させるという意味で、すごく良い問題だと思いました。この入試問題について、代ゼミの解答速報ウェブサイトには、「北大らしい手で数える力が試される問題」と書かれています。北大らしいとはなんのことかわからないですけれど。　今はもうNTTに戻っている、未来ねっと研究所の井

上さんという研究員の方がいたのですが、その方がこの入試問題をすぐにGraphillionで解いていました。大学入試問題をこれで解くのも大人気ないのですが、ここに示したような非常にコンパクトな記述だけでこの問題が解けます。

現在、電力網への応用にも取り組んでいます。ちょうどわれわれのERATOプロジェクトが始まって２年目に東日本大震災がありまして、今後長期的に電力が不足した時に、それを自然エネルギーで補うためには、電力網の解析･制御技術が必須だろうと考えました。もともと震災前からも取り組んでいたのですが、情報系の研究者として貢献したい想いもあり、ERATOプロジェクトでの取り組みを加速させています。これは、配電網で変電所から各町内に送電する時の最適な繋ぎ方を探す問題なのですが、スイッチの数に対して膨大な数の繋ぎ方があるので、それを列挙して網羅的に調べるには、先ほどのおねえさんの問題と非常に似た方法で高速に解く必要がありました。この問題のベンチマークとして使われている標準的な配電網の例題では、スイッチの数が468個あるのですが、正しく配電できる仕方というのは、2000那由他通りも存在します。これを１の桁まで網羅的に精密に調べて、そのなかで損失が最小になるような組合せを探すことに成功しており、さらに現在この問題を数10分ほどで計算できるような技術をつくっています。このあたりについては、現在も早稲田大学の先生と共同でいろいろな話しをしています。また、そのほかにもさまざまな実問題に取り組んでいます。避難所を配置する配置問題も、じつは配電の区割りの問題と非常によく似ている。それから、日本の民主主義を支える重要な問題である選挙区の区割りの問題も最近やっています。これも、区割りを列挙してその中から実現可能な区割りを探します。

NTTの研究所とも協力関係がありまして、現在CS研から研究員の安田さんに来てもらっていまして、電力網の解析やGraphillionの開発の応用をやってもらっています。離散構造処理の産業界への貢献を目指す目的で、NTT研究所だけでなく、産業界のいろいろなメーカーの方とも共同研究をしています。最後は少し駆け足になりましたが、「離散構造」とはなにかということと、「BDD／ZDD」の話、それから「フカシギの数え方」にまつわるさまざまな実社会への応用のお話しをさせていただきました。どうもありがとうございました。