Title	コンテンツネットワーク形成におけるメタデータの限界
Sub Title	
Author	金子, 晋丈(Kaneko, Kunitake)
Publisher	慶應義塾大学デジタルメディア・コンテンツ統合研究センター
Publication year	2019
Jtitle	慶應義塾大学DMC紀要 (DMC review Keio University). Vol.6, No.1 (2019. 3) ,p.29- 46
JaLC DOI	
Abstract	
Notes	特集 DMC研究センターシンポジウム第8回「デジタル知の文化的普及と深化に向けて」メタデータ再考 開催日時 : 2018年11月20日 (火) 14:00 ~ 17:30 開催場所 : 慶應義塾大学日吉キャンパス西別館1 講演
Genre	Departmental Bulletin Paper
URL	https://koara.lib.keio.ac.jp/xoonips/modules/xoonips/detail.php?koara_id=KO32002001-00000006-0029

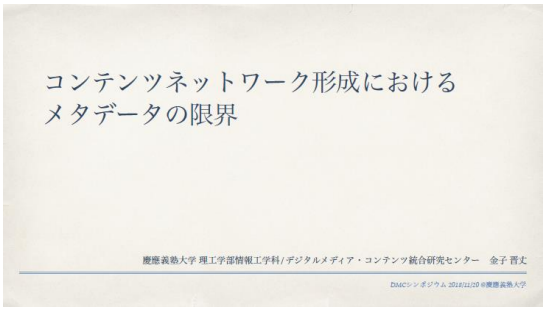
講演

「コンテンツネットワーク形成における メタデータの限界」

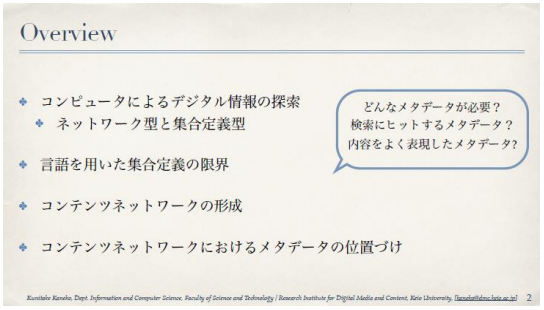
金子晋丈

（慶應義塾大学理工学部専任講師

・DMC 研究センター研究員）

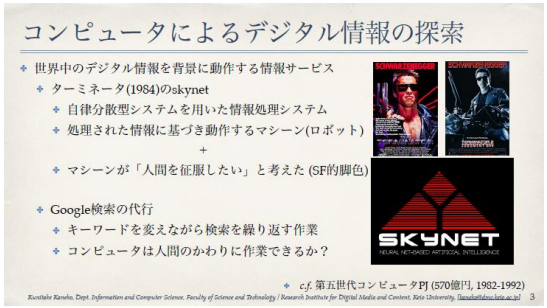


ご紹介ありがとうございました。理工学部情報工学科の金子と申します。DMC 研究センターの研究員も務めております。きょうはコンテンツネットワークとメタデータに関してお話ししたいと思います。ここにいらっしゃる方はコンテンツネットワークとはなんだと思われていると思うのですが、そのあたりも含めてお話しできればと思っております。きょうの内容ですが、コンピューターによるデジタル情報の探索はどういうプロセスを踏むのだろう、人間でもそうですが、情報の探索のフェーズみたいなところを、最初お話しできればと思っています。



現実にいまどういう形で探索をやっていて、本当はあるべき姿とはどういうものなのかなというところから始めて、現在のキーワード検索に代表される言語を用いたやり方の限界が伝わればいいかなと思っております。一方で、「ネットワーク型と集合定義型」と書いていますが、ネットワーク型のほうがコンテンツネットワークというものなのですが、それはどういうふうにしたらできていくのだろうとか、最終的にはメタデータはどういうものなのかというものをお話しできればと考えております。

まず、コンピューターによるデジタル情報の探索ですが、こちらにある映画のタイトルを持ってきました。『ターミネーター』、ご存じない方はいらっしゃらないと思いますが、『ターミネーター』の最初のものが左側、2 個目が右側、その後続編も出ていますが、スペースがないので二つだけです。



最初のものが出されたのが 1984 年です。僕はつい最近までこの 1 は見たことがなかったのですが、『ターミネーター』のストーリーは皆さんご存じですか。漠然と覚えていらっしゃる方もいると思いますが、マシンというロボットとコンピューターが一体化したようなものが人間と戦っていくという話です。その悪の権化みたいなやつが skynet というもので、ニューラルネットベースの artificial intelligence が動いています。これはべつに僕が付け加えたわけではなくて、これが当時から言われていたトピックです。ですから、皆さんが最近 AI と言っているのは、1984 年の話をしているということになります。

1984 年はどういう時代かと考えると、たとえば国内では第五世代コンピュータプロジェクトというのがありました。知っている方は知っていると思いますが、調べましたが、莫大な 570 億円を 10 年間かけて使い、人工知能ベースのコンピューターをつくるという新しいプロジェクトです。ですから、時期がドンピシャであるというのを、まず一つ分かっていただけると思います。

もう一つが、インターネットです。インターネットというのは、自律分散型のシステムで動くと。その二つを組み合わせて、『ターミネーター』というのは『アバター』をつくったジェームズ・キャメロンという監督がつくっているのですが、ジェームズ・キャメロンはたぶんここの二つを組み合わせて、このストーリーを組んでいるのではないかと、僕は現段階では思っています。

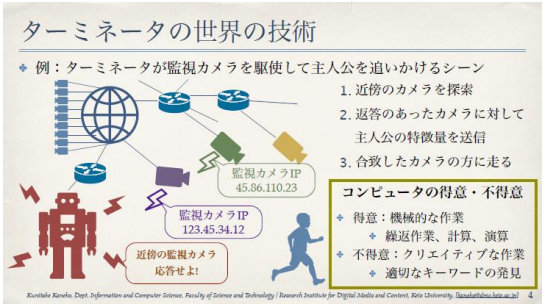
整理しますと、自律分散型インターネットにそれぞれが独自のルールに基づいて動きながら、全体として協調的な動作をしていく。自律分散型システムを用いた情報処理システムと、処理された情報に基づき動作するマシン、ロボット、それが技術的な背景にあり、SF 的な味付けとして、マシンが人間を征服したいと考え始めたということで、この『ターミネーター』はストーリーとしては動いていることになります。

もちろん誤解なく言っておきますと、これは SF 的脚色ですので、べつにコンピューターが頑張ったところで、マシンが人間

を征服したいと考え始めることはないので、これを議論すればするほど世の中が破滅に向かうのかとか、金子の研究は軍事研究だとか、全然そういうわけではないのでご安心ください。

こういった『ターミネーター』の世界が、1984 年にある種 SF として出てきたにもかかわらず、一向にコンピューターが独自に情報を解析して、それを使っていくという形にまだまだなってきていないのはなぜかというと、技術的には考える必要があるかなと思います。

これが 1984 年の例ですが、いまのわれわれの生活でいくと、Google 検索の代行をできるコンピューターがあるかというのと、ほぼほぼ似ている考え方になると思います。皆さんが Google 検索をやるときは、キーワードを入れて結果が返ってくる。少し不満のある、所望の結果が返ってこなかった場合は、キーワードを変えてもう一度入れて結果が返ってくる、それを何回か繰り返すという作業が Google の検索作業になると思うのですが、これはコンピューターに代行させることができません。なぜかというと、コンピューターは次に浮かぶキーワードが浮かんでこないからです。これがそもそもの上の話（ターミネータの skynet）と下の話（Google 検索の代行）で実は共通しているバックグラウンドということになっています。



さて、技術者ですので『ターミネーター』の世界を実現しようと思ったら、どんな IT システムがあればできるのかを考えるわけです。SF を見ながら、「これはどういう技術があったらできるかな、これは無理だね」ということを考えるのは楽しいのですが、『ターミネーター』のリアルなやつを持ってくるとあれかなと思いまして、ロボットにして、これは逃げている主人公みたいな絵です。

すごく典型的なシーンとして、スライド 4 に、監視カメラが街中にあって、世界中に監視カメラがインターネットにつながって存在しているみたいな絵を描いています。では、『ターミネーター』のシーンでいくと、マシンが逃げている人を探し出すためにはなにをしなくてはいけないかと。よくイメ

ージ的には監視カメラを捉えて、あそこに映っているというのが分かって、あちらにいるんだと追いかけるシーンになっていると思いますが、それをやるためにはものすごく大変なことが必要なんです。

なぜかというと、ロボットはこの人が映っているカメラがどこにあるかを知らない。そもそも自分の周りにどんなネットワークカメラがあって、それが本当に自分の近くにいるかが分からないので、仮にたとえば世界中に 1 億台の監視カメラがあったとして、1 万台でもいいですが、その中から自分の近辺もしくは逃げている主人公の近辺のカメラを、10 台でも 100 台でも限定してチョイスすることができないので難しくなってきます。

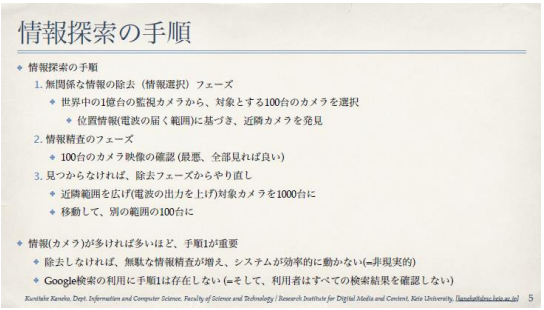
ネットワーク的には違う位置にそれぞれいます。この監視カメラはこんな IP アドレス（45.86.110.23）、このカメラはこんな IP アドレス（123.45.34.12）、違う場所にあって、違う場所にあるから、ただ単に IP アドレスの一番下の桁を 1 増やせばいいとか、1 減らせばいいとか、そういう話ではなくて、なにかしらのメカニズムでこのカメラを見つけてこないといけないというところに、最初の技術ポイントがあります。

ところが、映画ではこんな話はなくて、自動的に見つかったみたいな、「やばい、やばい」と思わされるわけですが、見つけてくることになります。一番僕が安直に思い付いた技術的な解決策は、ロボットが自分から電波を発して、近くの監視カメラがそれに対して応答するわけです。電波が届く範囲にいる監視カメラが「僕の IP アドレスなんだよ」というのを返してあげる。そうすると、このロボットは自分の近くにある監視カメラの一覧を入手できるので、それに基づいてその監視カメラたちに、「こういう特徴の顔をしたやつがいたら教えろよ」みたいなことを言うと、『ターミネーター』の世界が出来上がってくるかなと思うわけです。なんとなく伝わりましたか。

コンピューターには得意なことと不得意なことがあるわけです。ここを理解しないとうまくシステムがつくれないわけですが、コンピューターが得意なのは完全機械的な作業です。繰返作業やこういう計算をしなさい、こういう演算をしなさいというのがコンピューターの得意なこと。不得意なことは、なにか新しい発想を求めること、これはコンピューターには現段階ではできないものになっています。

これを考慮してこんなシステムをつくっ

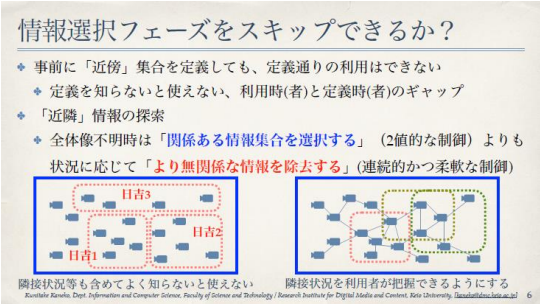
てみたわけですが、これをもう少し整理してみます。これが情報探索の手順の一つのサンプルと思って整理しているわけですが、最初はその 1 億台なり 1 万台なりの監視カメラから、追いかける相手が映っているであろう、映っているかもしれないエリアの 100 台を選んでくるフェーズ、逆に言えば 1 億台から 100 台以外を消すフェーズが最初にあります。その次に出てくるのが、その 100 台のカメラ映像から顔が一致するものを見つけてくる、情報を精査するフェーズになる。100 台で見つからなかったら、もう一度 100 台を 1000 台に変えて探し直しましょうというのがだいたいの流れになります。



こう考えたときに、いまの検索技術はなんなんだろうとか、いまわれわれがやっている情報の探索はどういうことなんだろうというので、たとえば 1 番の情報選択フェーズをカットするとなにが起こるのだろうか。1 億台世の中にあるので、仮に 1 億台のリストがあったとしたら、1 億台にすべて問い合わせて、その顔が映っていますかというのを返す。それだけ聞くと、それでもいいのではないかと思うかもしれないですが、そういうことをしたいユーザーが 1000 人いたら、1 万人いたら、10 万人いたら、そのシステムは動くのだろうか。そうすると、それは動かないわけです。リクエストが集中して処理ができないので、動かないということになります。

では、皆さんがやられている Google 検索は、この例でいうと、実は最初から情報の精査のフェーズに入ってきます。キーワードを入れてこれに合致するものだけを返してくださいということです。たとえば、Keio University で Google 検索すると何件ヒットするか分からないですが、軽く 1 万を超える数字がヒットしましたみたいなものが出てくると思います。皆さんが見るのは最初の 10 個ぐらいです。それは精査していないことになりますね。1 万件ヒットしたうちの 100 件だけ、最初の 10 件だけを見ると、残りの 99 万 9990 件は無視されるということで、実は適切な情報探索のフェーズを経ていない、情報探索ができていないというのが、実際のいまの IT における情報探索の問題と考えています。

最初の情報選択、ここには絶対あいつはいないというのを排除するフェーズをスキップできるかという問題は、いろいろ考えさせられる問題です。それを二つのやり方で対比的に書いております。



たとえば、『ターミネーター』の世界で、頭の中はアメリカのハリウッドのような感じになっているところで、日吉というすごくローカルな地名を出してあれなのですが、たとえば日吉に監視カメラがあって、この監視カメラ群は日吉１丁目にあります、この監視カメラ群は日吉 2 丁目にあります、この監視カメラは日吉３丁目にありますとあらかじめ定義してあげて、それを使って逃げ続ける主人公をうまく見つけられるかという話をここでは取り上げたいと思います。

たとえば、逃げる相手がここだけでぐるぐる回っていて、逃げているのが日吉１丁目だというのが分かれば、なんとなくオーケーそうです。でも、逃げているときに、日吉３丁目のほうに行ったり２丁目になったり、実際日吉１丁目と日吉３丁目は隣接していないのですが、架空の話だと思っていただいて。ぐるぐるやっていると、いまその主人公は何丁目にいるのだろうか、あの通りを隔てると日吉何丁目に行くんだろうか、そういったことを全部把握した上で、いまどこの集合に対して検索をかけないといけないのかということをやらないといけないのが、実はこちらのモデル（スライド 6 の左の図）になっています。

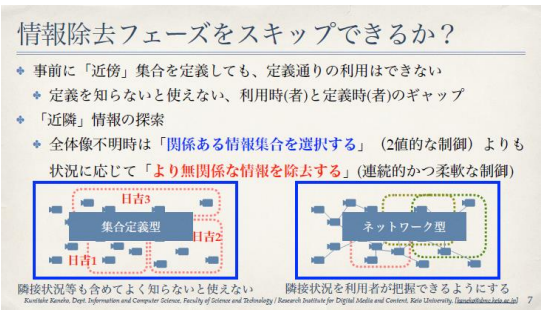
こちらに別のモデル（同、右の図）を書いてみました。こういう集合を捉えるのではなく、それぞれのカメラが相互に近いカメラのことを知っているだけの世界です。この世界だと１カ所場所が決まって、いまここに映ったよとなると、ユーザーはそこから近い位置にいるので、その近傍のカメラを自在に設定することができます。仮に 100 台のカメラでいなかったら、1000 台に広げてあげましょうとか、ユーザーが移動したらユーザーが移動した方向でまた円を描いてあげましょうという形で、変幻自在にその集合を変えてあげることができるのが、こちらのネットワーク型のモデル（同、右の図）になっています。

すなわち、ここで言いたかったのは、こういう形態（同、左の図）を取るときは、関係ある情報の集合が相互にどういうつながりになっていて、どのエンティティがどこに属しているのかを完全に把握している

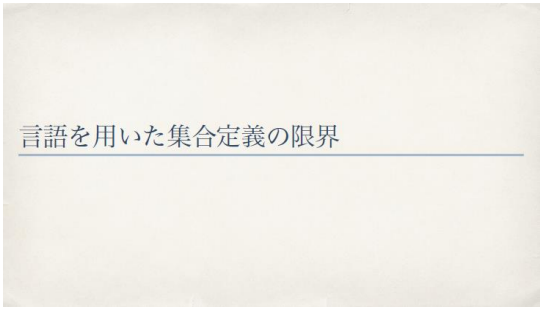
場合には、非常に速やかに場所を特定できて、情報を選択することができるのですが、よく分からないときは実はこれはうまく使えないことになります。いまそもそも自分は日吉 1 丁目にいるか 2 丁目にいるかすら分からない場合には、左のモデルはなかなか使えないということになるかと思います。

言いたかったのは、たとえば Google 検索をするときに、皆さんは世界中のボキャブラリーを知っていて、それぞれがどういう位置で集合が管理されていて、それが分かっているのであれば、キーワードを入れて適切に情報を持ってくることができるけれども、それを知らないのであれば、うまく使えないということをこの例は表しています。

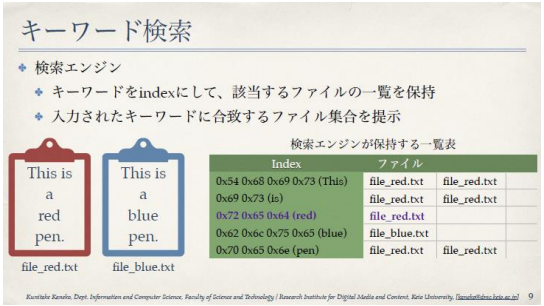
どちらかというと、世界中の情報を知らないのであれば、こちらのモデル（同、右の図）のほうがより適していて、自分が少なくとも知っているところから徐々にたどって、本当に知りたいところに近づいていく、そういうことができるのがこの右のモデルになろうかと思います。



以下では、この左のほうを集合定義型、右側をネットワーク型という名前で呼びたいと思います。



次に、集合定義型で言語を用いて集合を定義した場合に、さらにどんな問題が加わってくるのかについてお話ししたいと思います。ここは先ほどの原先生のお話と微妙にオーバーラップする部分ですが、たぶん原先生のほうが非常に具体的な事例を挙げてご説明いただいていて、僕は少し抽象的な話になっているかもしれないので、うまく頭の中で merge させながら聞いていただきたいと思います。

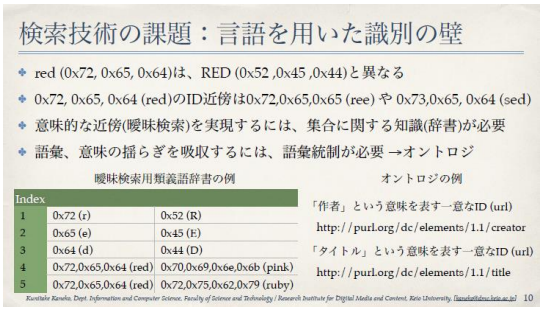


非常にシンプルなキーワード検索の実現方法を書いております。キーワード検索をやるときには、たとえば file_red.txt と file_blue.txt があったときに、それぞれの要素である単語がそれぞれ表に管理されることになります。「This」というのは This

という意味ではなくて、実際コードになって落ちていきますので、0x54、0x68、0x69、0x73、これが This です。ですから、This のコンピューター的な識別は「This」ではなくて「0x54、0x68、0x69、0x73」、この数字の列によって一意に決まってくるということになります。「This is a red pen.」「This is a blue pen.」、これを両方全部分解していくとこういう形になって、それぞれの数字の列が含まれているファイルが一覧表になると、キーワード検索のベースが出来上がることになります。

すなわち誰かが「This」という検索キーワードを入れると、入力された文字列が 0x54、0x68、0x69、0x73 だったら、これだねと言ってこの二つが検索結果に返るし、「red」の 0x72、0x65、0x64 というのが与えられると、この file_red.txt が返ってくるというのが検索の基本的な動きです。

あえてここでこの数字の列を、分かりにくいにもかかわらず出したのには理由があります。それは、たとえば小文字全部の「red」と大文字全部の「RED」は数字としては違うからです。「0x72、0x65、0x64」と「0x52、0x45、0x44」というのはどう見ても似つかない。red と RED だから一緒だと、それは意味の世界であって、コンピューターの世界では完全に数字の列で判断しますので、それは全く違うものということになります。



よりコンピューターの立場に立って考えると、0x72、0x65、0x64 に近いものはなにか、すごくコンピューター的に簡単に近いものを出すと、0x72、0x65、0x65、1 ビット違いですが「ree」です。全然 red とは違います。もしくは、red の r のところだけが変わって 0x73、0x65、0x64 で「set」。こういうふうに ID 近傍、機械的にすぐプログラムができて近くを見つけるというのは、意味的にはまったく異なるところを指していることになります。

そうすると、コンピューター的な数字の列でなにかしようというと難しいので、意味的な近傍もしくは曖昧検索ともいいますが、それを実現するには集合に関する知識（辞書）が必要になってくるということになります。一番端的な例ですと、先ほどの red と RED は同じだよということをやりたければ、72 が入力されたら 52 にも変換して検索しなさいとか、65 が入力されたら 45 にも変換して出しなさいというテーブルを持つわけです。そうすると、大文字、小文字を区別なく検索することができるようになります。

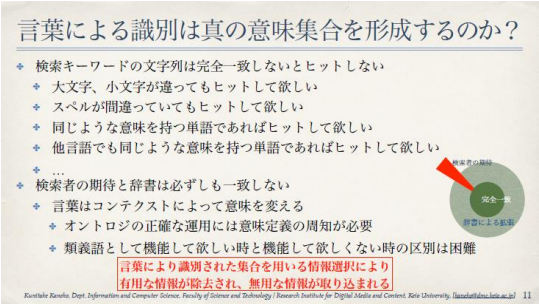
さらには、もっと高度な曖昧検索をしたくて、赤みがかった色を言うときに、赤と言うのか、ピンクと言うのか、朱色と言うのか、いろいろあると思うのですが、そういったものの広がりを検索で実現しようと思うと、このテーブルに全部書かないといけないことになります。0x72、0x65、0x64 は 0x70、0x69、0x6e、0x6b、これは pink ですが、変換してそれも探しなさいとか、同じように red が入ったら ruby となっていますが、0x72、0x75、0x62、0x79 で ruby の色を探しなさいということをやるわけです。

こういうテーブルを持ち始めると、なにが起こるかというと、いまは赤色を探したいから red と pink でいいかもしれないですが、red と pink を区別して調べたいときには、これは邪魔にしかならないわけです。もしくは、Red と ruby を区別して用いたいときには邪魔にしかならない。それが全部一緒くたになってテーブルに入ってくるので、こういったものはなかなか難しいということになってきます。大文字、小文字ぐらいはまだ、大文字、小文字を区別して検索するのはよくありますよね、これをやっているだけです。それ以上のものが入ってこないというのは、弊害も大きいからということになります。

こういう意味の広がりをどういうふうに対処するか、言葉のばらつきをどういうふうに対処するかというので、一つ出てくるのが語彙を統制する。Red というとこういう意味だということを定義するわけです。先ほどの原先生の話でも出てきましたが、たとえばオントロジーとして「作者」という意味があります。作者という意味が一意な ID として url がこれで指定されるのですが、先ほど「作者はいろいろな定義がありますから、熟読してくださいね」とさらっと流されましたが、まさに熟読して、それを理解して、それとたがわぬ使い方をしない限り、実はこういうものはうまく動かないということになります。

言葉による識別というのは、本当に真の意味集合を構築できるのかというのが、そもそも疑問に挙がってきます。利用者からすると、検索キーワードの文字列は完全に一致しなくても見つけてほしいという気持ちになると思います。僕だってなります。スペルミスをしたら直してよとか、大文字、小文字はもちろんのこと、同じような意味を持っているのだったら、それも探してくれたらいいじゃないかと思います。もしくは、日本語で出てこない情報だったら、英訳してくれてそれも探してくれたら幸せだよねと思うのですが、ベースはキーワードの文字列は完全一致でないと駄目ですよと。文字とその文字コードというところで全然違ってきますので、ちゃんと ID 化されるとそれが区別されてしまうという世界で、

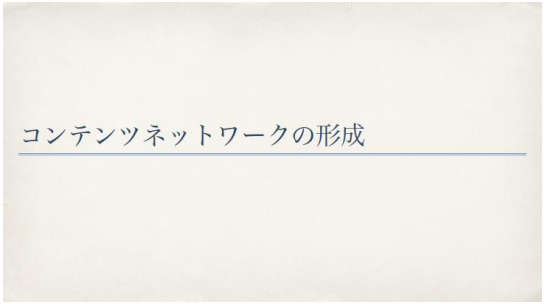
こういったミックスはなかなか難しいことになります。



スライド 11 に絵にしてみましたが、検索者の期待というものがあって、完全一致で出てくる部分、辞書によって先ほどのマッピングのテーブルで出てくる部分、たぶんユーザーが期待している部分。イメージ、すごく期待している部分のほうが辞書による拡張よりももっと先に行っていて、辞書によって、拡張によって得られたのは少しというところが現実的なところなのかなと思います。

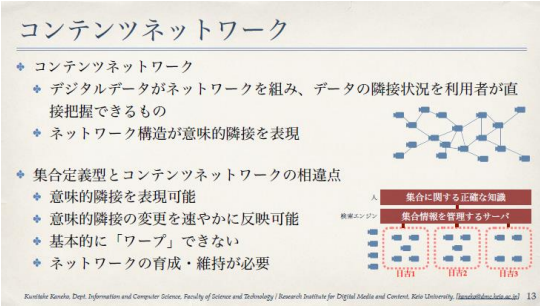
それは、そもそも言葉はコンテクストによって意味が変わってきますし、オントロジーの正確な運用には意味の定義の周知が必要であり、類義語として定義するとしても、それを機能させたいとき、させたくないときのコントロールが非常に難しいです。コンピューターは人間の状況を把握できないので、それをつかむことは非常に困難になるかと思います。結果として、言葉で集合を定義するとなにが起こるかというと、有用な情報が除去されて、無用な情報が取り込まれるというのが言葉による識別の難しいところなのかなと考えています。

ここまでのところで二つお話をしました。情報を選択してくるというときに、集合を定義するとフレキシビリティがないという話と、その集合が言葉によって定義された場合、そこに曖昧性という言葉のコンテクストへの依存性があって、うまく機能しないということになります。



さて一方で、もう一つ対抗馬として出したネットワーク型についてお話ししたいと思います。コンテンツネットワークの形成と書いていますが、ネットワーク型の話です。

いろいろ出てきますが、コンテンツネットワークとはなにかというと、デジタルデータがネットワークを組むと。データの隣接状況を利用者が直接把握できると書いていますが、言ってみれば先ほどの『ターミネーター』のカメラとカメラが相互につながっているイメージ。カメラが情報に置き換わっていくと思っていただけるといいかなと思います。



ある情報がある情報とつながっている、この情報はより自分に近いような位置にいる、もしくは遠い位置にいる、そういったものがこのグラフ情報によって分かってくると。ホップが離れていくと遠い、ホップが近いということになります。すなわち、ネットワーク構造というのはグラフ構造と思っていただいてもいいですが、意味的隣接を表現可能だというのが一つ大きな特徴になっています。

このネットワーク型を集合定義型の先ほどの例と比較していくと、いくつか違いがあります。先ほど言った意味的隣接を表現可能だということが一つと、意味的隣接が少し変わったら、その部分だけ変えてあげれば、それを使う人はそのときになって反映された状況を使えますということ。

あとは、一方でワープができないです。ワープができるというのはどういうことかというと、たとえば日吉１丁目でグループ化される、集合として定義されている、このカメラはクリックで切り替えられるような集合だと思うのですが、それはこのネットワーク的にはここととここに対応しているかもしれない、ここととここになっているかもしれない。でも、これはネットワークがつながっていないので、直接にはそこには行けないということになります。

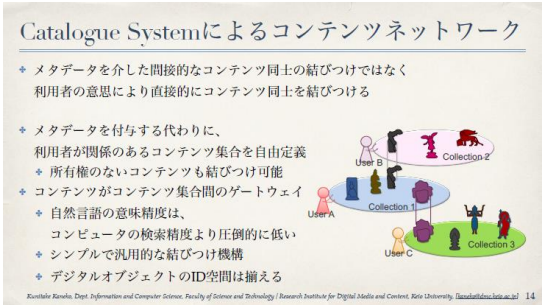
ネットワーク構造を取った場合は、そのネットワークの上でしか通っていくことができないので、離れたところにジャンプはできないことになります。でも、頑張って誰かたどった人がいて、これとこれが実際近いことが分かったら、ここに線を引いてあげるだけで、そのあとの人はワープできるというのがネットワーク型の特徴になります。

こうやって話をしていると、ネットワーク型はイケてるよねと思っていただけたら僕はうれしいのですが、一番難しい点があります。コンセプトは非常にシンプルで、近いものをグラフで表していくというすごくシンプルな構造なのですが、大変なのがスライド 13 の下に書いた、ネットワークの育成と維持が必要になります。なぜ難しいかというと、ネットワークにはメカトーフの法則というものがあります。これはネットワークの価値が n の 2 乗に比例するというものです。

一番シンプルな例でいきますと、いまどきは携帯電話だと思いますが、携帯でも携帯ではなくても電話があるとしたときに、世界中に 1 台しか電話がなくて、それを所有していることに意味はまったくないわけです。誰にもかからないし、意味のないものです。2 人だけが持っていたら、その 2 人だけがコミュニケーションできるので、その 2 人の関係が親密であれば非常に有効に使えます。でも、それだけです。ホットラインみたいなものです。3 人になってくると、3 人がそれぞれ話せるようになってくるので、2 人だけしか持っていないときよりも便利だよねと。4 人、5 人、6 人、7 人とどんどん増えていけばいくほど、その価値は上がっていくということになります。

ですから､ネットワークサービス全般に、たとえば Facebook もそうですが、あれは 5 人でやっていても全然つまらないと思うんです。でも、あれだけのスケールで動いているから新しい情報がやってきておもしろいよね、みんなやっているなら僕もやろうとなるわけです。これがネットワーク技術です。

インターネットも同じです。3 台のパソコンがインターネットにつながっています。3 台だけでインターネットが構成されている世界。「べつにそれは要らないのでは」とみんな思うと思うんです。でも、世界中のネットワーク、世界中のコンピューターが同じネットワークにつながっているからこそ、インターネットをつないでおかなきゃとなるわけです。これが先ほどのメカトーフの法則という考え方で、そこのしきい値を超えるまでが、実はネットワーク型システムの一番難しいところになります。



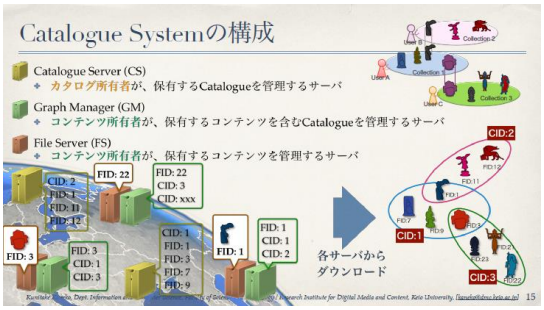
スライド 14、うちの DMC でやっている研究の紹介をさせてください。うちのネットワーク型のシステムです。うちはネットワーク型でシステムをつくるというチャレンジをしています。一般的なメタデータで物事を見つけてくる検索のようなやり方、それはメタデータを介した先ほどの言葉で表されたつながりです。同じ ID を持っている、共有しているという間接的なコンテンツ同士の結び付けではなくて、利用者の意思により直接的にコンテンツ同士を結び

付けると。

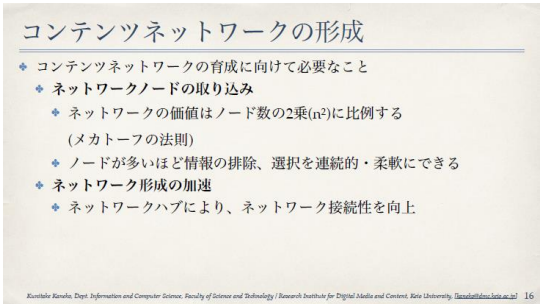
たとえばAさんがこのコンテンツとこのコンテンツは関係あるよねと思ったら、ここで1個決めてあげる、結び付ける、線を引いていると思っていただいても構いません。メタデータを付与する代わりに利用者が自由に定義してあげましょうと。言語化せずにただ単に集合を与えるだけ、もしくはリンクをつないであげるというのが基本的なシステムの考え方です。

これはあとで出てきますが、所有権、自分が持っていなくてもこのリンクをつくれるというので、実はネットワーク育成を促進させようとしています。コンテンツが、たとえばこの黒いオブジェクトを見ていただくと、User B も User A も Collection にそれぞれ入れている。このコンテンツ同士が自動的につながるメカニズムを Catalogue System は持っています。単独で Collection だけをつくっていたらネットワークは分離した世界なので、まったく使い勝手が悪いのですが、同じコンテンツを持つと、そこが自動的につながるメカニズムを導入することで、ネットワークが構成されるようになっているというのが Catalogue System です。シンプルで簡単に結び付けられるというのをキーにした技術になっています。

具体的な方法を話しても、頭がこんがらがるといけないので簡単に説明しますが、先ほどの上の User A、B、C がそれぞれ自分で関係をつけたグループをつくった、もしくはネットワークをつくった。それぞれが LAN をつくると思っていただいても構いません。



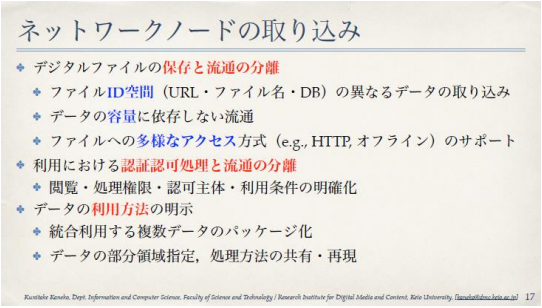
それが実際のデータとしては、左のような形で入って、CID:1、CID:2、CID:3 のような形でそれぞれネットワークが組まれることになっているのが Catalogue System の構成になっています。詳しい話は技術展示のところで聞いていただければ、説明してもらえると思います。



さて、コンテンツネットワークをつくっていこうとしたときに、難しいことが二つあります。一つはノードの数をいかに増やしていくかという話と、その増やしたノードの間にネットワークをどう組ませるかというところが、コンテンツネットワーク形

成においては非常に重要なポイントになってきます。

ネットワークノードが増えるほうがいいというのは、先ほど話したことです。プラス、ノードが多くなってくると、情報の粒度、どのぐらいまでの情報が欲しいかというバリエーションがたくさん出てきますので、ユーザーに応じて適切な情報を選択することができるという意味でも、もちろんノードが増えれば増えるほどいいかなということになります。ネットワーク形成の加速ですが、ハブをあちこちに置こうね、そうするとネットワークにつなぎやすくなるでしょうというのが、簡単な理解になります。



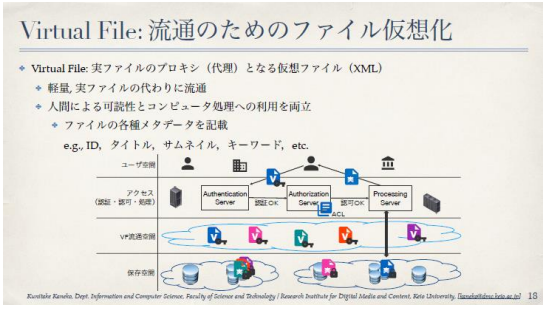
まず、ネットワークノードの取り込みですが、どうやったらこの関係のあるネットワークにノードを持ってこられるか、どんどんファイルを増やしてこられるかというのをいろいろ考えました。それが実際の設計としてこういうふうに落ちてくるのですが、まず保存されているファイルをそのまま流通させることはやめようと。保存する話と流通する話、そういったファイルがあるという話は完全に切り離しましょうというのが一つ目の話です。

なぜかというと、あるファイルは URL が HTTP でアクセスできるところにあるけれども、あるファイルはオンラインにないかもしれない。オンラインに上げておくのは非常に管理の手間がかかるので、そういうのはないかもしれない。もしくは、それがデータベースに入っているかもしれない。それを透過にアクセスしないといけないとなると、全部オンラインに上げてくださいとか、ものすごくシステムの負荷が上がってきます。こういった別々の場所に保存されているものを、きちんとハンドリングしたいので、まず切り分けましょうというのが一つ目。

二つ目が、最近だとデジカメやスマホで撮っても軽く 5MB とかいくような時代に、5MB、10MB のファイルをスパンスパンとやりとりして幸せかというと、スパンスパン行けばいいですが、行かないので不快なわけです。そうすると、データの容量に依存しないような形でやるためにも保存と流通は分離しましょうと。あとは、先ほども言いましたが、HTTP、オフラインのサポート等いろいろあるのを、どんな形でもできるようにするためにも、流通にはそういうエンティティは要るけれども、実際のファイルはどこかに保存されていてもいいみたいなことを言っています。

あとは、ファイルを使うときに課金したいとか、誰だったら使っていいみたいな話が出てきます。そういったところもファイルがあるよとか、このファイルは便利だよという話とは別のところで管理しましょうと。お金を取る、誰が見られないようなのと、実際にそういうのがあって便利だよという話は別だということです。

あとは、最後にデータをどう利用できるのか、先ほども少し出てきましたが、こういうフォーマットで書かれていて、どういう情報が含まれているか分からないと使えないので、そういう利用方法を明示したり、ある誰かがつくったコンテンツの一部だけを流通させたかったら、そういう部分指定もできるようにしたりするというのが、ネットワークノードの取り込みの話です。



いまのコンセプトを全部突っ込んでつくっているのが、スライド 18 の Virtual File という話で、あとでここで技術展示のポスターを学生さんがやっていますので見てください。保存の話、Virtual File といいますが、その流通の話、認証の話、実際使う話、先ほどの Catalogue System を使う話はもっと上にいるのですが、それを切り離して動かすようにしましょうと。

流通したときに、これは一種のチラシみたいなものです。実際のサービスではなくてチラシみたいなもので、チラシにどんな情報を載せておけばアトラクティブなのかみたいなことは考えないといけないので、ここにメタデータがいろいろ載ってくるのかなと考えています。たとえば、タイトルがなにとか、サムネイルも載っていてとか、最低限の流通情報が Virtual File には記載されるのかなと考えています。

もう一つ、ネットワーク形成の加速ですが、ただノードがネットワーク上に存在するだけでは、ネットワークの価値は上がってこなくて、ネットワーク化される、要するにどんどんいろいろなところとつながってくることによって、あちこちに行けるようになるわけです。

まず、システムとして一番肝というか重要視しているのが、デジタルコンテンツを所有していなくてもネットワークの線を引けるということです。インターネットの世界で極端な話、たとえば僕は CNN をよく見ますと。CNN を見るためには日米間の回線が太くないといけないといった場合、日米間の回線を太くするみたいなことを自分が思っていないにもかかわらずやるということです。それができることで、ネットワークがどんどん出来上がってくるという

ことです。日米間のと言っても、日米間の既存のラインの横に引くのではなくて、ショートカットで行けるようなものをつくってしまうということです。

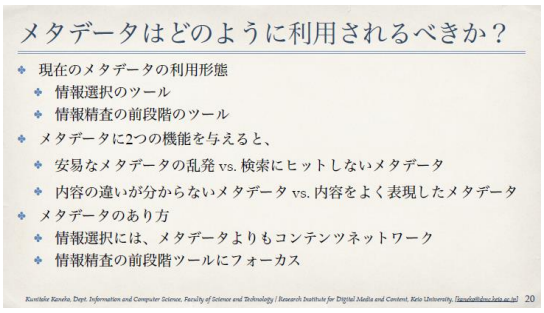
もう一つ、別のアイデアでネットワーク形成を加速しようとしているのですが、それがハブを用意するという考え方です。三つのやり方をスライド 19 に書いています。モノハブ、位置ハブ、イベントハブです。

モノハブはその名のとおり、同じモノに対して、たとえばこれですと、「伊右衛門 500ml ペットボトル」というモノを、違うデジタルファイルに、いま四つになっています。同じモノを三つのデジタルファイルが共有していることになります。あるときはこれを A さんがつくり、あるときは B さんがつくり、それは全然別の場所で行われるかもしれないけど、工業品なのでなかなか難しいですが、違う場所でつくられて、違う人によってつくられたかもしれないけれども、こういったモノがハブ機能になるというのは、ネットワークを形成するのに便利かなというのが一つ目です。

二つ目が、位置を軸にいろいろなものがつながってくると。たとえば、慶應だと日吉の記念館をいま建て替え工事中です。下は古いやつ、上は新しいやつ、これは想像ですが、それがこの場所でリンクしてくると、別の人が同じ場所で体験したものに飛んでいけることになります。

最後のイベントハブというのが時間軸です。たとえば、去年の DMC シンポジウムでこんなのがあったよと、Twitter のハッシュタグみたいなものですが、違う人がそれぞれ撮った写真であっても、それが同じイベントでそこをハブにいろいろ広がっていけるというような工夫をすることで、ネットワーク形成が加速されるかなと考えております。

さて、いままで集合定義型の話、集合定義型が言語、言葉で表現されたときの課題、一方でネットワーク型、ネットワーク型の課題とその解決の仕方をお話ししてまいりました。ここで改めてメタデータはどのように利用されるべきではないかというところに立ち返りたいと思います。



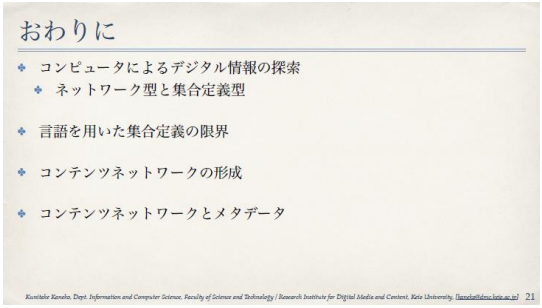
言うまでもなく、集合定義型、言語による集合定義というのがいまのキーワード検

索であり、ほぼほぼいまのメタデータ検索のやり方になっていると思います。現在のメタデータの利用形態は、先ほどの情報探索の手順の 1 番と 2 番を両方やっているのではないかと思います。すなわち、情報選択のツールであり、たくさんの情報リソースの中からそれを選んでくる、1 億個から 100 個を選んでくるための道具であり、そしてその 100 個の中から、いったいなにが書かれているのか中を見極める、でも本文までは読みたくないみたいなときに使われる情報精査の前段階のツール、この二つが混在しているのがいまのメタデータの利用のされ方かなと思っています。

しかしながら、この二つの機能をメタデータに同時に与えてしまうと、なにが起こるかというと、情報選択のツールだと思うと安易なメタデータを乱発し始めると。引っ掛かってくれなかったら困るといって、なんでもかんでも、富士山が小さくでも写っていたら富士山、富士山と入れるということです。でも一方で、情報精査の前段階のツールだと思って入れたキーワードは、すごく精緻なキーワードであるかもしれないけど、決してほかの人が思い付かないみたいな話、これは相反する、対立するような軸かと思います。

もう一つが、メタデータは与えたけど、こちらの写真に与えられたメタデータとこちらの写真に与えられたメタデータは全く同じだよね、それは本当に中身を表しているのかということです。一方で、きちんとその違いが分かるようなメタデータ、そういったところがこの二つの混在によってミックスしてくることになります。

僕が思うに、情報選択というのはメタデータを使うのではなくて、コンテンツネットワークを使うべき、そこで情報の選択をして、情報の選択をしたあとにより精査するときに、そのプロセスを簡略化する、もしくは簡便にするためにメタデータがあればいいのかなと思っているという状況です。



最後に「おわりに」ですが、コンピューターによるデジタル情報の探索はどんなシステムでできるのかな、どんな手順になるのかなというところからスタートして、メタデータ、既存のキーワード検索の限界というものを伝えたつもりです。一方で、ネットワーク型にすることで、柔軟、フレキシブルでいろいろなユーザーの状況に応じた探索をすることができるというのがネットワーク型で、ネットワークの形成の仕方、そして最後にメタデータが現状とどのような関係であるべきかというところをお話し

いたしました。

金子 晋丈（かねこ くにたけ）

慶應義塾大学理工学部専任講師・DMC 研究センター研究員。専門はアプリケーション指向ネットワーキング。特に、デジタルデータの利活用を促すデジタルデータのネットワーク化について研究を行っている。2001 年東京大学卒業。2006 年同大学院情報理工学系研究科博士課程終了、博士（情報理?学)。同大学院新領域創成科学研究科での特任助教を経て、2006 年 9 月より慶應義塾大学デジタルメディア・コンテンツ総合研究機構、特別研究助教。2007 年、同機構特別研究講師。2012 年 4 月より現職、デジタルメディア・コンテンツ統合研究センター研究員を兼任。