Can a Computationally Creative System Create Itself? Creative Artefacts and Creative Processes

Diarmuid P. O'Donoghue, James Power, Sian O'Briain, Feng Dong*, Aidan Mooney, Donny Hurley, Yalemisew Abgaz, Charles Markham

National University of Ireland Maynooth, Ireland,
*University of Bedfordshire, UK.

Motivation

- From the ICCC 2014 CFP

> **High Level Issues**
>
> Papers which, in part or fully, address high-level general issues in Computational Creativity are particularly welcome, including notions such as:
>
> …
>
> Process **vs.** product: addressing the issue of evaluating/estimating creativity (or progress towards it) in computational systems through study of what they produce, what they do and combinations thereof.. …

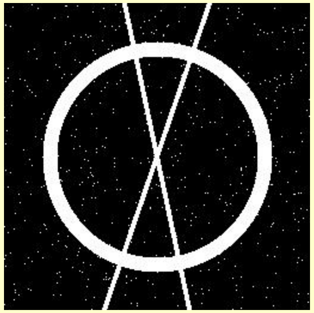
- Stereotype is: artistic **artefacts** *vs* scientific **process**

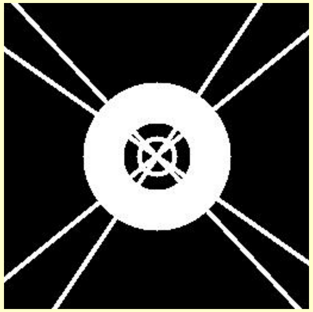
Introduction

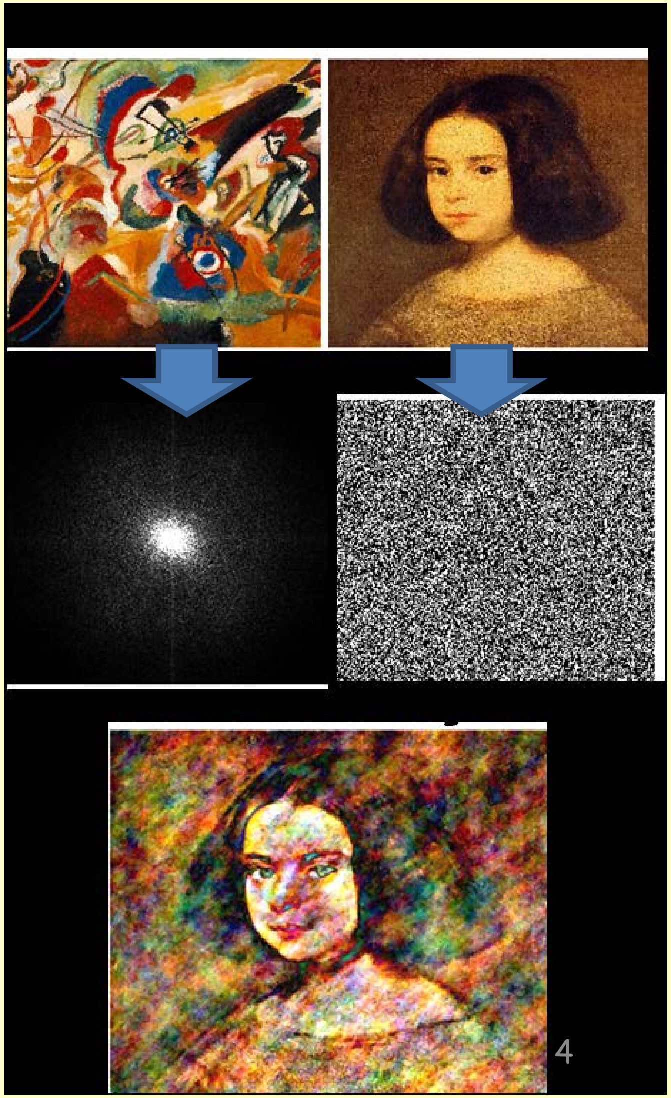
1. Process vs Product Creativity
   - ImageBlender, RegExEvolver
2. 2D Matrix of Knowledge and Process
   - Using educational attainment theory
3. Levels of Creativity
   - Inspired by Turing machines
4. Summary/Conclusion

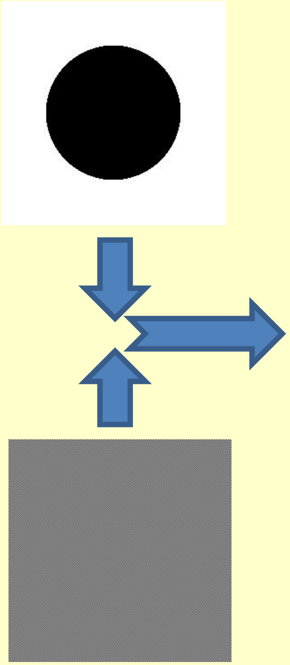
1. *ImageBlender*

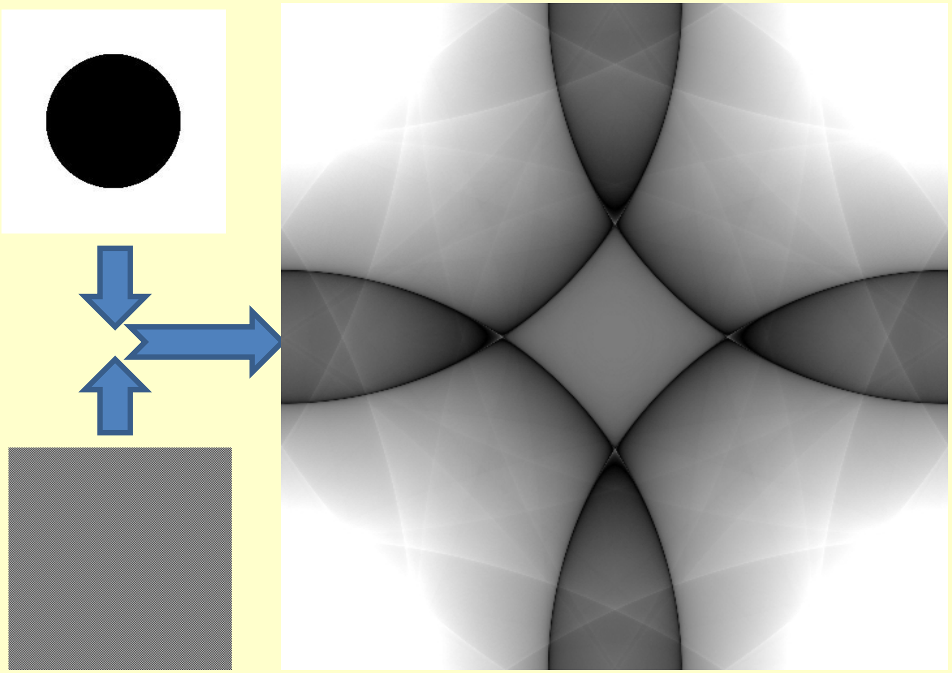
- ImageBlender blends FFT of images
  - phase & frequency
- General multi-objective evolutionary algorithm
  - Evolved filters (below)

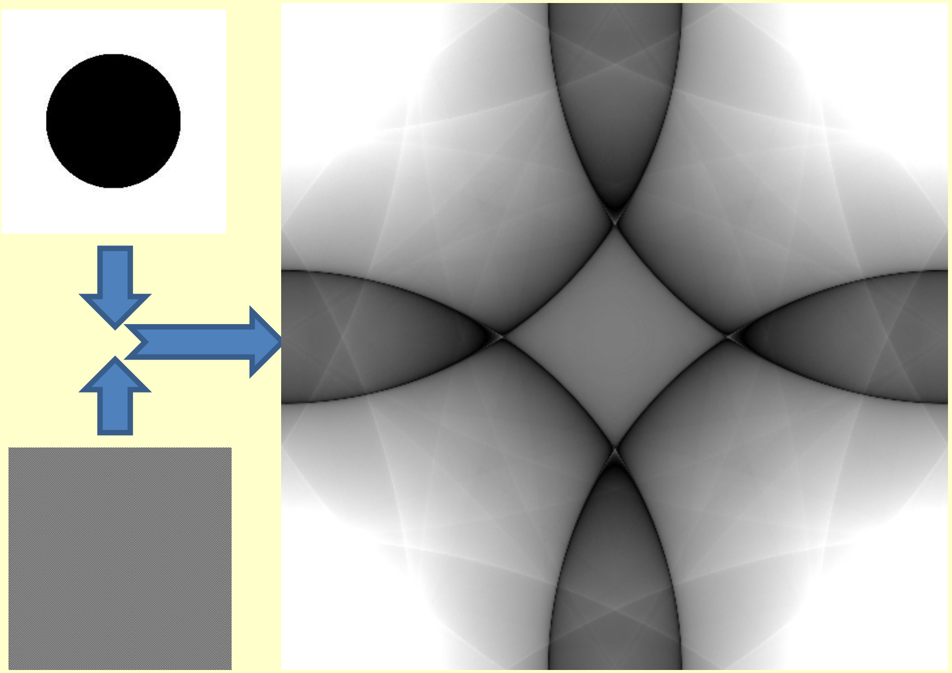


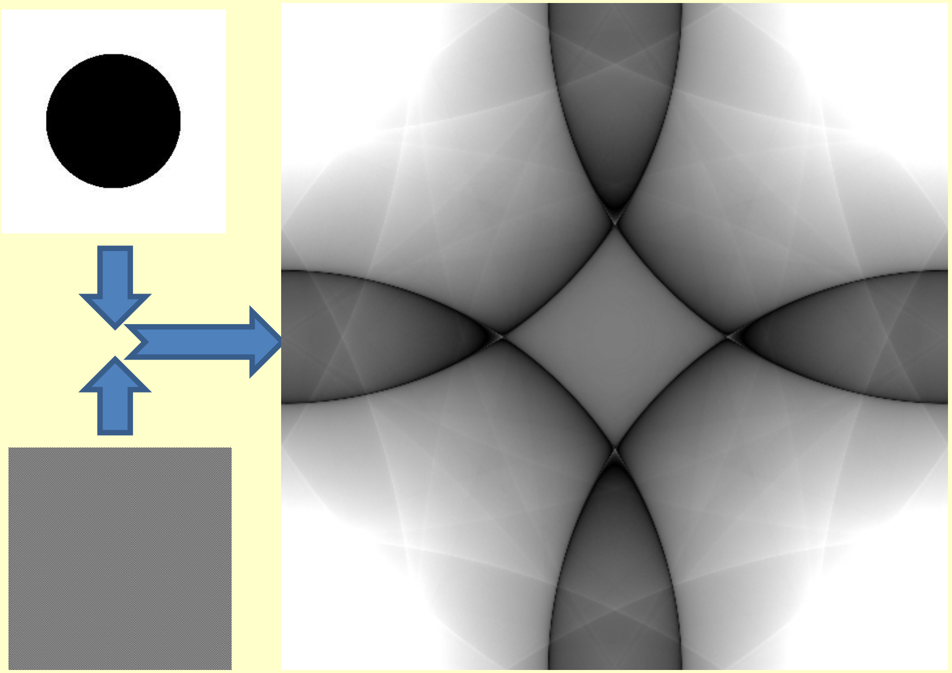


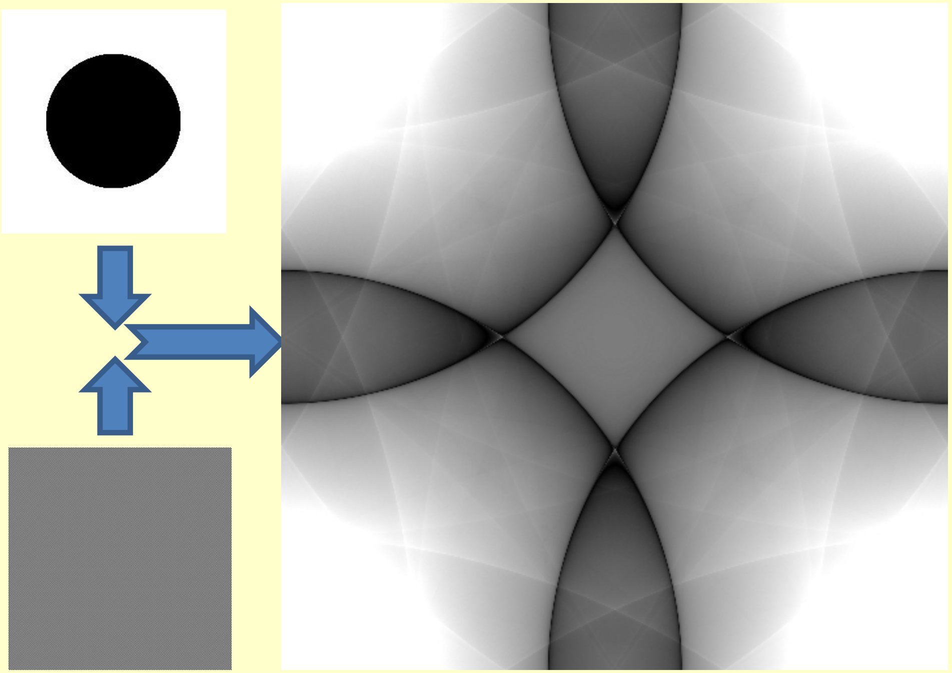


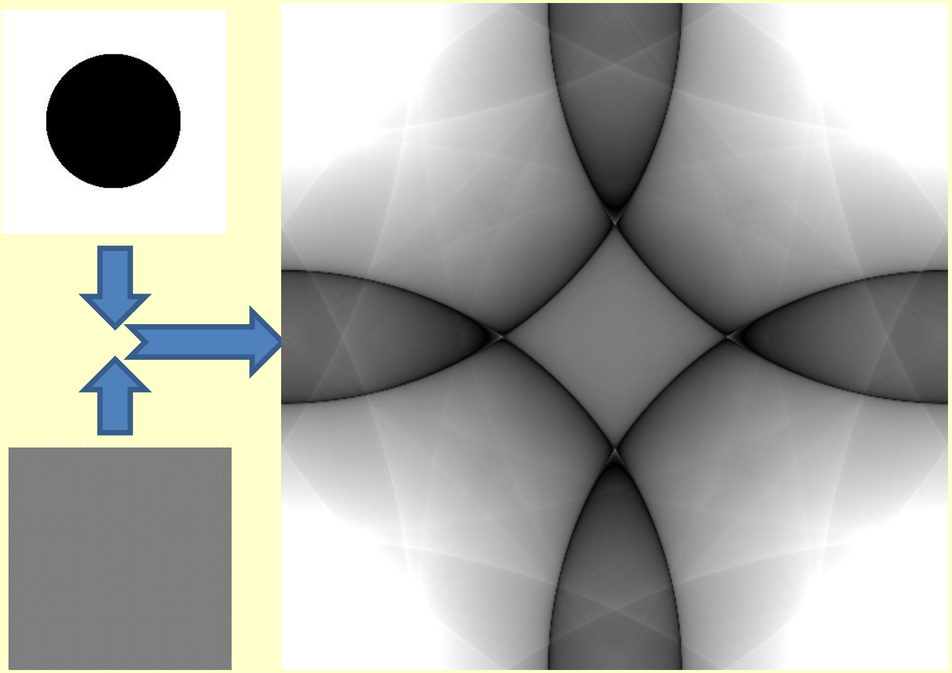


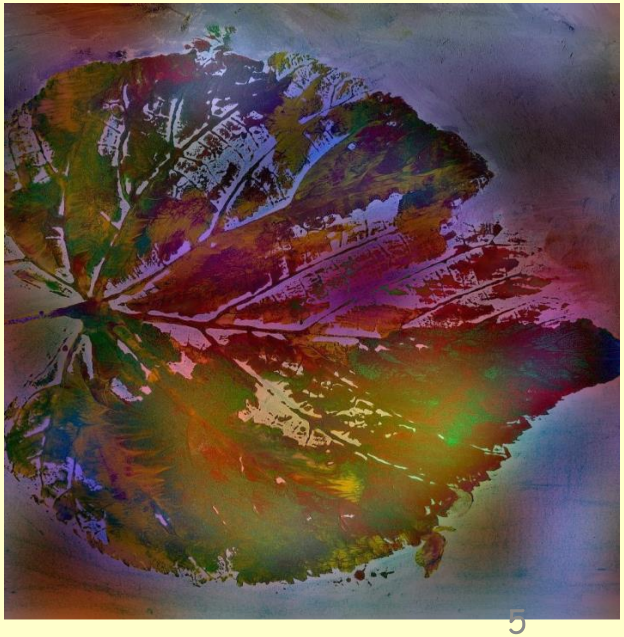












Regular Expression & *RegExEvolver*

- Create a new RegEx, using another RegEx as its inspiration
  - Reg. Expr. being a simple Turing Machine
  - General evolutionary algorithm, multi-objective
- Potential application to software testing
  - create positive and negative test cases
  - ImageBlender & RegExEvolver are guided by the complexity/interestingness of their outputs

ImageBlender and RegExEvolver

- Both are multi-objective evolutionary algorithms
  - Small input sets, make "minimal" assumptions about the creative domain
  - Both estimate "interestingness" , serving as one of their objective functions
- Some similarity and dissimilarity with original inputs are other objectives
  - for novelty & usefulness
  - But can we compare them in non-mechanistic terms?

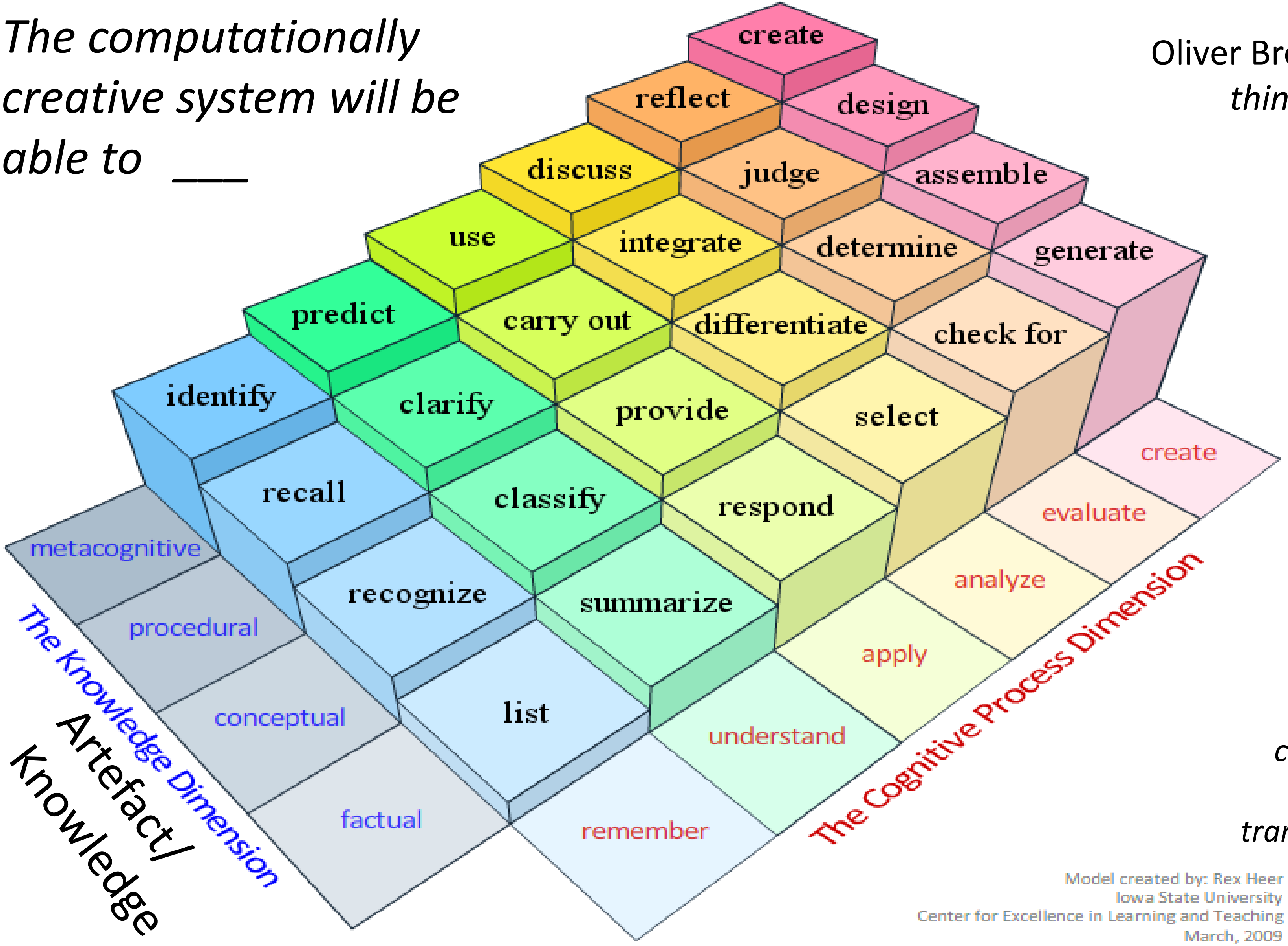
2. Educational Attainment

- Use an education theory as a reference framework for resolving tension between *artefacts* & *processes*
- Bloom's *Revised* Taxonomy values creativity within educational systems
- But D. Krathwohl's 2D matrix provides a more useful perspective
  - Distinguishes between **Knowledge** and Cognitive **Process**

Educational Attainment Analogy

The computationally creative system will be able to ___

Oliver Brown *"created things & creative people"*



	remember	understand	apply	analyze	evaluate	create
metacognitive	identify	predict	use	discuss	reflect	create
procedural	recall	clarify	carry out	integrate	judge	design
conceptual	recognize	classify	provide	differentiate	determine	assemble
factual	list	summarize	respond	select	check for	generate

Model created by: Rex Heer
Iowa State University
Center for Excellence in Learning and Teaching
March, 2009

Artefact/ Knowledge

In addition to *combinatorial, exploratory, transformational*

Levels of Creativity

- Not creative: Bottom of the matrix
- Approaching creative: middle of the matrix
  - Apply/procedure (*carry out*)
  - Evolutionary algorithms, Analogical reasoning
- Create is both a Process Dimension and a level of attainment
- Create/Factual (*generate*) can be creative
  - *New* Mersenne Primes, *ImageBlender*
- Conceptual/Create (*assemble concepts*)
  - RegExEvolver

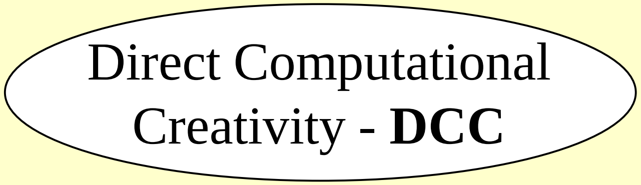
Levels of creativity

- Higher levels of creativity
  - Evaluate/meta-cognitive knowledge
  - Design a creative procedure…
- Peak of educational attainment
  - Create/meta-cognitive process
  - Note: this model requires the creation of meta cognitive knowedge for "true" creativity
- **But**: Is that the highest level possible for computational creativity?

3. Other Levels of Computational Creativity

- …remaining focused on *artefacts* and *processes*?

other than replacing the Regular Expressions in RegExEvolver with higher levels of the Chomsky hierarchy

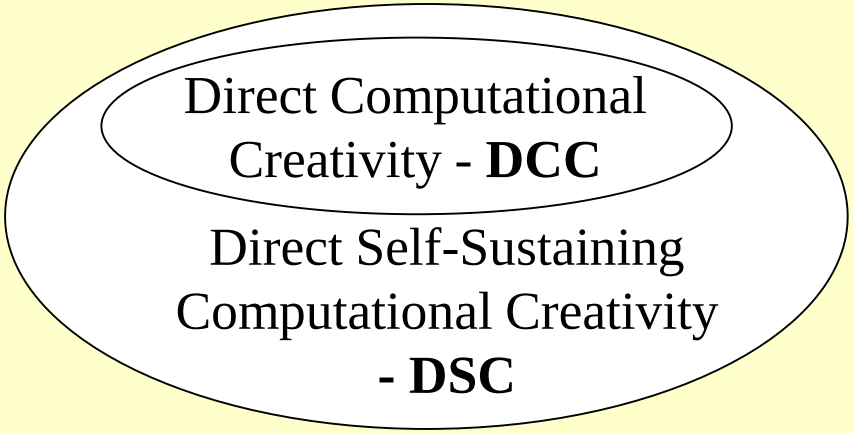


Direct Computational Creativity - **DCC**

3. Other Levels of Computational Creativity

- …remaining focused on *artefacts* and *processes*?

other than replacing the Regular Expressions in RegExEvolver with higher levels of the Chomsky hierarchy



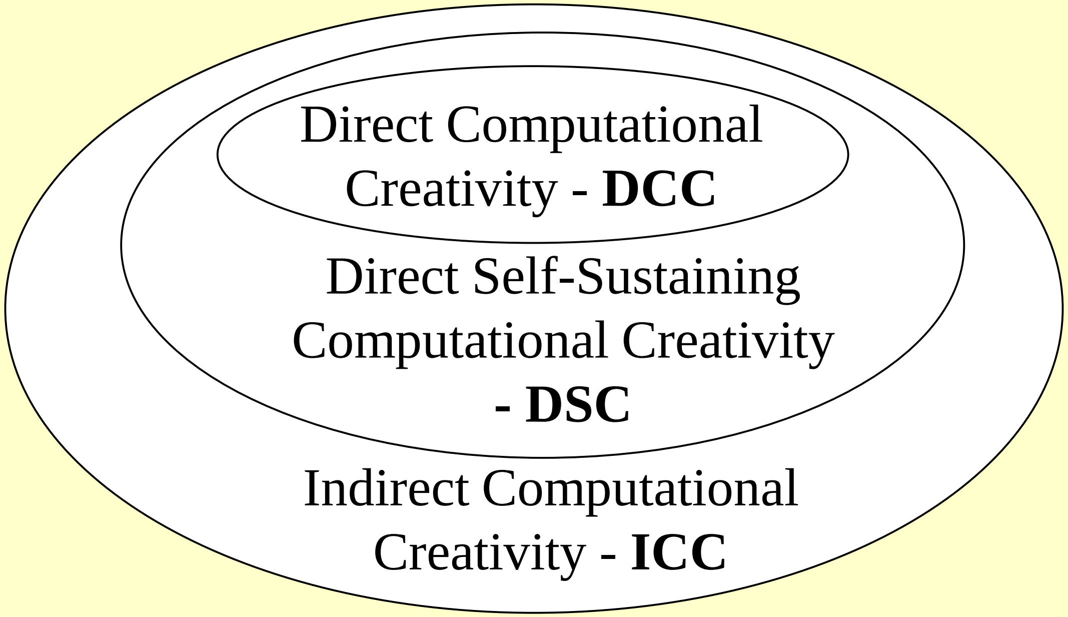
Direct Computational Creativity - **DCC**

Direct Self-Sustaining Computational Creativity **- DSC**

3. Other Levels of Computational Creativity

- …remaining focused on *artefacts* and *processes*?

other than replacing the Regular Expressions in RegExEvolver with higher levels of the Chomsky hierarchy



Direct Computational Creativity - **DCC**

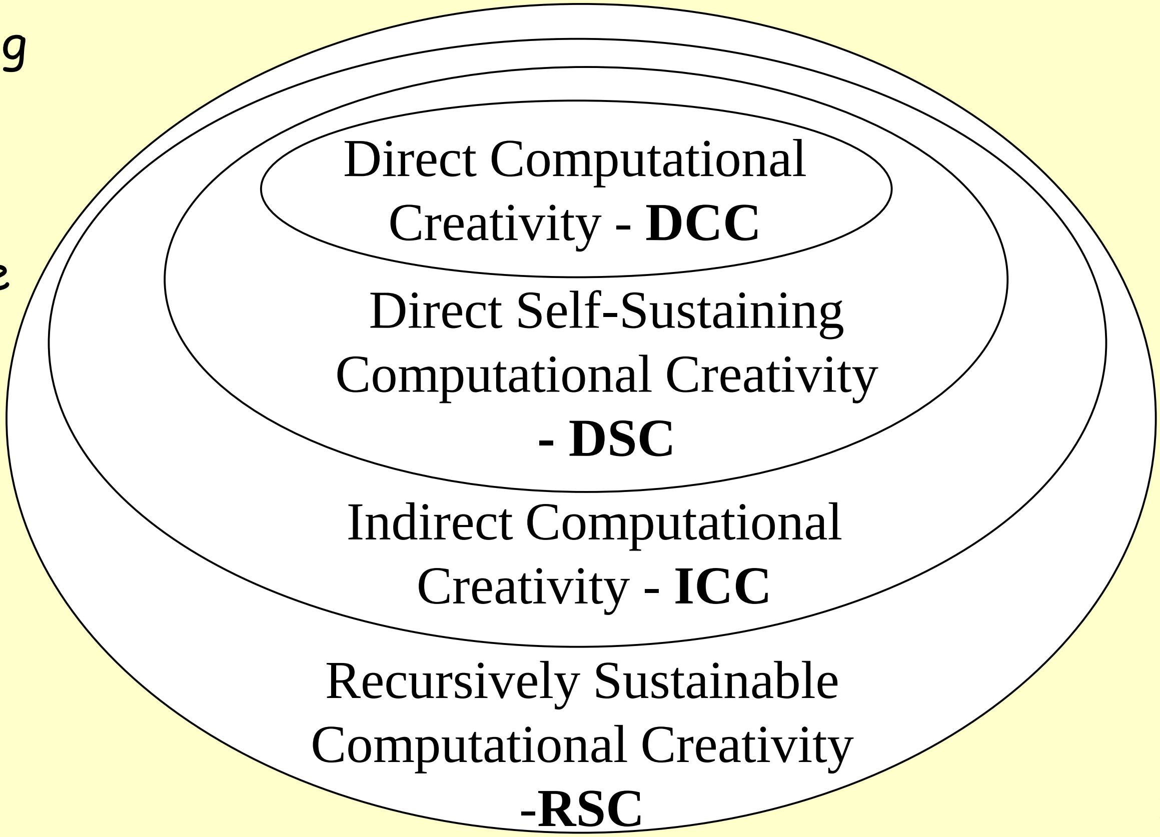
Direct Self-Sustaining Computational Creativity - **DSC**

Indirect Computational Creativity - **ICC**

3. Other Levels of Computational Creativity

- …remaining focused on *artefacts* and *processes*?

other than replacing the Regular Expressions in RegExEvolver with higher levels of the Chomsky hierarchy



Direct Computational Creativity - **DCC**

Direct Self-Sustaining Computational Creativity **- DSC**

Indirect Computational Creativity - **ICC**

Recursively Sustainable Computational Creativity **-RSC**

Hierarchy of Creative Outputs

1. Direct Computational Creativity (DCC):
   - A process producing creative artefacts
   - *ImageBlender* and *RegExEvolver*
2. Direct Self-Sustaining Creativity (DSC):
   - Creative outputs serve to **drive** subsequent creativity, perhaps via reflection
   - Even beyond *regular creativity* (Gardner, 1993)

Hierarchy of Creative Outputs

3. Indirect Computational Creativity (ICC):
   - output is a creative process and that creative process is itself creative
4. Recursively Sustainable Creativity (RSC):
   - the created process itself creates processes that are at the level of RSC

4. Summary/Conclusion

- We described two evolutionary models of creativity (*ImageBlender, RegExEvolver*)
- Krathwohl's 2D Matrix provides a useful reference framework to compare *artefact* and *process* centred creativity
  - But meta-cognition necessary for true creativity (in this framework)
- Presented a 4-level Hierarch of computational creativity
  - focused on interactions between creative artefacts and processes

Towards *Dr Inventor*: A Tool for Promoting Scientific Creativity

D.P. O'Donoghue[1], H Saggion[2], F. Dong[3], D. Hurley[1], Y. Abgaz[1], X. Zheng[3], O. Corcho[4], J.J. Zhang[5], J-M Careil[6], B. Mahdian[7], X. Zhao[8]

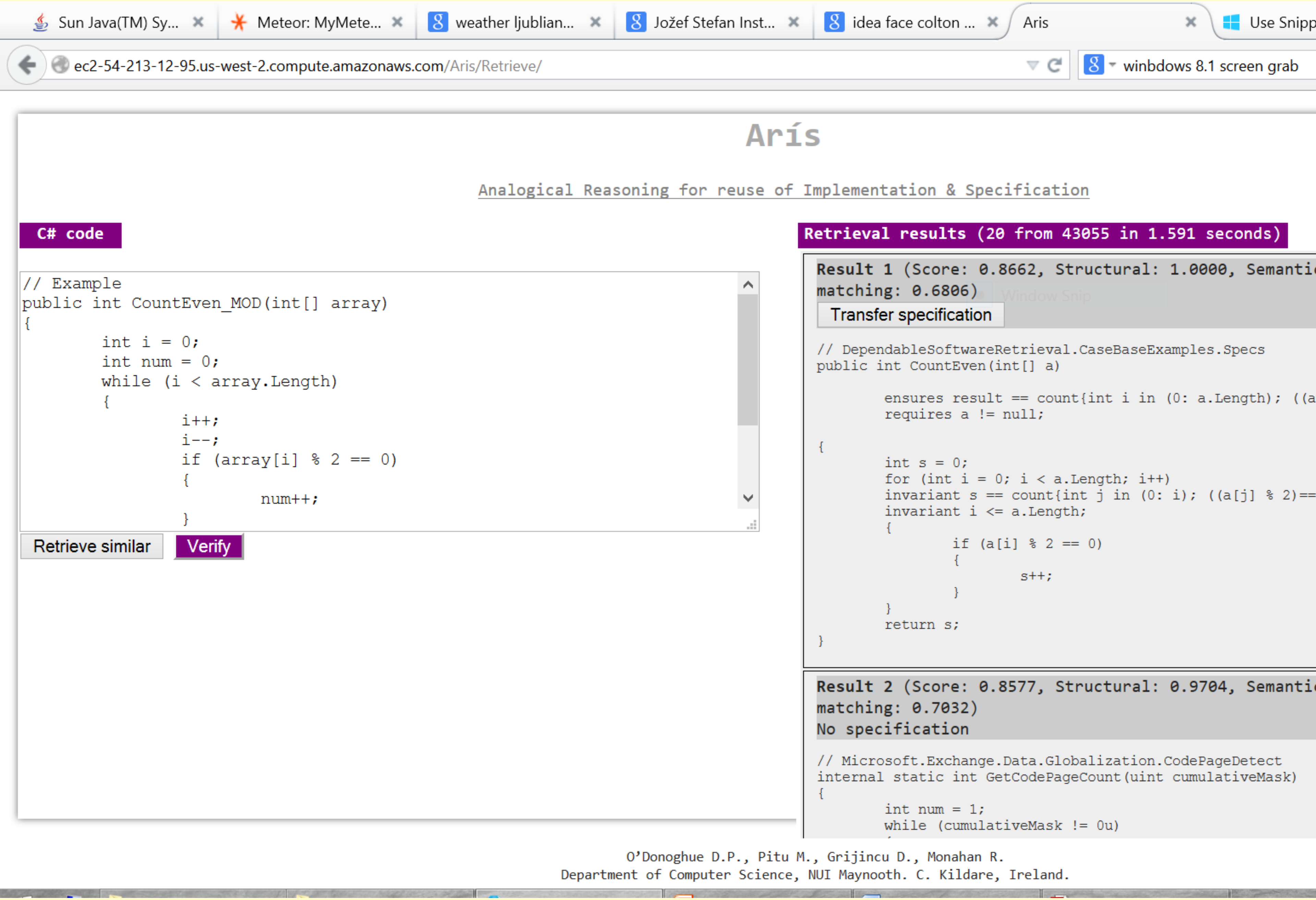
1 National University of Ireland Maynooth, Ireland.
2 Universitat Pompeu Fabra, Barcelona, Spain.
3 University of Bedfordshire, UK;
4 Universidad Politécnica de Madrid, Spain
5 Bournemouth University, UK
6 Intellixir, Manosque, France
7 ImageMetry, Prague, Czech Republic
8 Ansmart, Wembley, UK

Objective

- Supplement the creativity of practising scientists
  - Dr Inventor aims to become a personal research *assistant*
- Hopes to discover creative analogies (Koestler, '64; Brown, '03; Boden, '09).
- Aimed at *Big-C Creativity* (Gardner,'93), *H creativity* (Boden,'92)
- Look for radical transformations inspired by analogically similar but semantically distant concepts
  - (Gick and Holyoak, 1980; Thibodeau and Boroditsky, 2011).
  - Overcome limits of Kilaza Analogy discovery system (O'Donoghue & Keane, '12)

Hypothesis Discovery

- Based on published papers and related *research objects*
  - Patents and other resources
  - Broader scope than the Aris project (Analogical Reasoning for Implementations and Specifications)
- Dr Inventor is based on computational model of analogical reasoning
  - (Gentner '83, Keane *et al*, '94; Gentner & Forbus '11)



Arís

Analogical Reasoning for reuse of Implementation & Specification

C# code

```
// Example
public int CountEven_MOD(int[] array)
{
        int i = 0;
        int num = 0;
        while (i < array.Length)
        {
                i++;
                i--;
                if (array[i] % 2 == 0)
                {
                        num++;
                }
```

Retrieval results (20 from 43055 in 1.591 seconds)

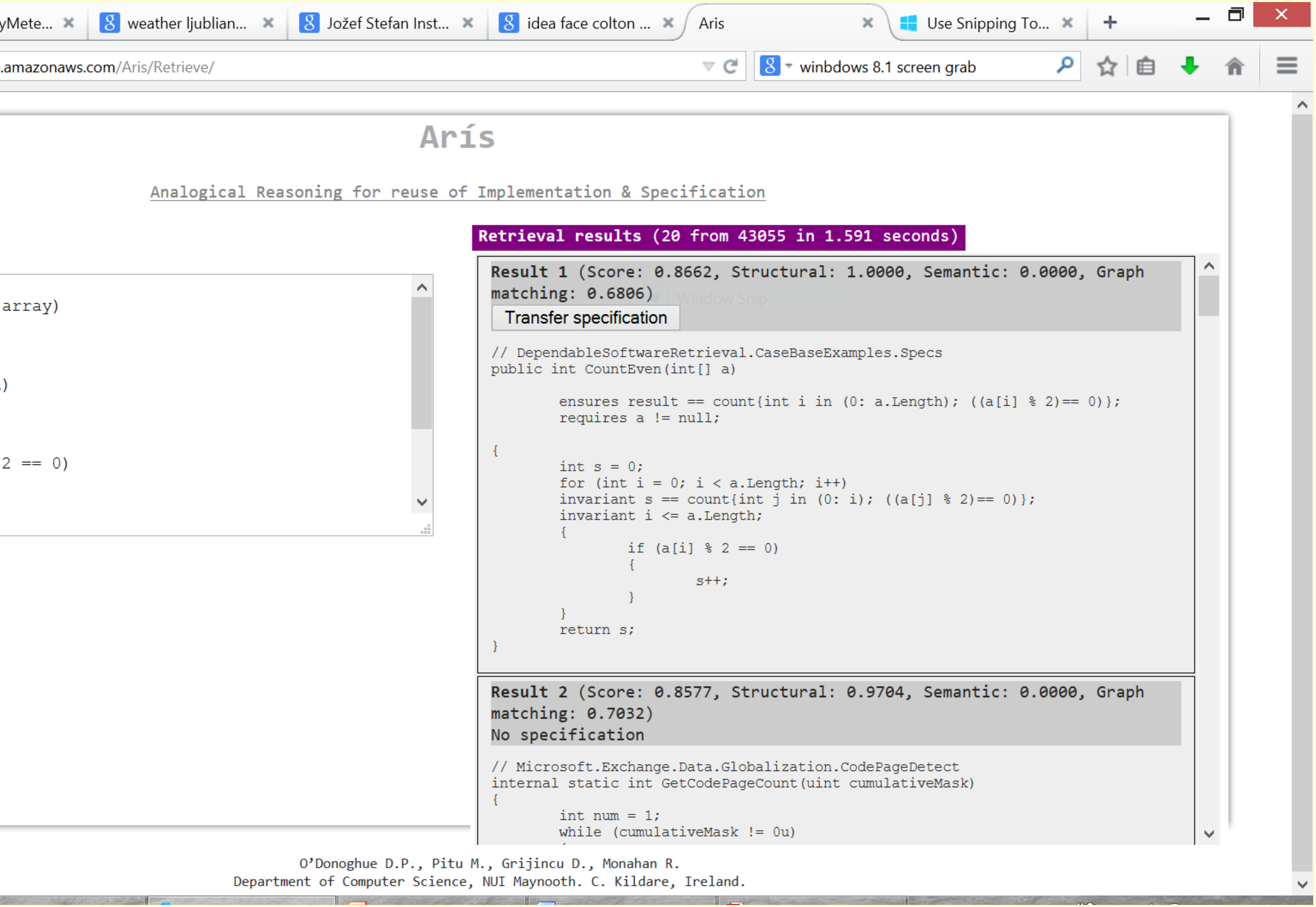
Result 1 (Score: 0.8662, Structural: 1.0000, Semanti
matching: 0.6806)

```
// DependableSoftwareRetrieval.CaseBaseExamples.Specs
public int CountEven(int[] a)

        ensures result == count{int i in (0: a.Length); ((a
        requires a != null;

{
        int s = 0;
        for (int i = 0; i < a.Length; i++)
        invariant s == count{int j in (0: i); ((a[j] % 2)==
        invariant i <= a.Length;
        {
                if (a[i] % 2 == 0)
                {
                        s++;
                }
        }
        return s;
}
```

Result 2 (Score: 0.8577, Structural: 0.9704, Semanti
matching: 0.7032)
No specification

```
// Microsoft.Exchange.Data.Globalization.CodePageDetect
internal static int GetCodePageCount(uint cumulativeMask)
{
        int num = 1;
        while (cumulativeMask != 0u)
```

O'Donoghue D.P., Pitu M., Grijincu D., Monahan R.
Department of Computer Science, NUI Maynooth. C. Kildare, Ireland.



Arís

Analogical Reasoning for reuse of Implementation & Specification

```
array)

)

2 == 0)
```

Retrieval results (20 from 43055 in 1.591 seconds)

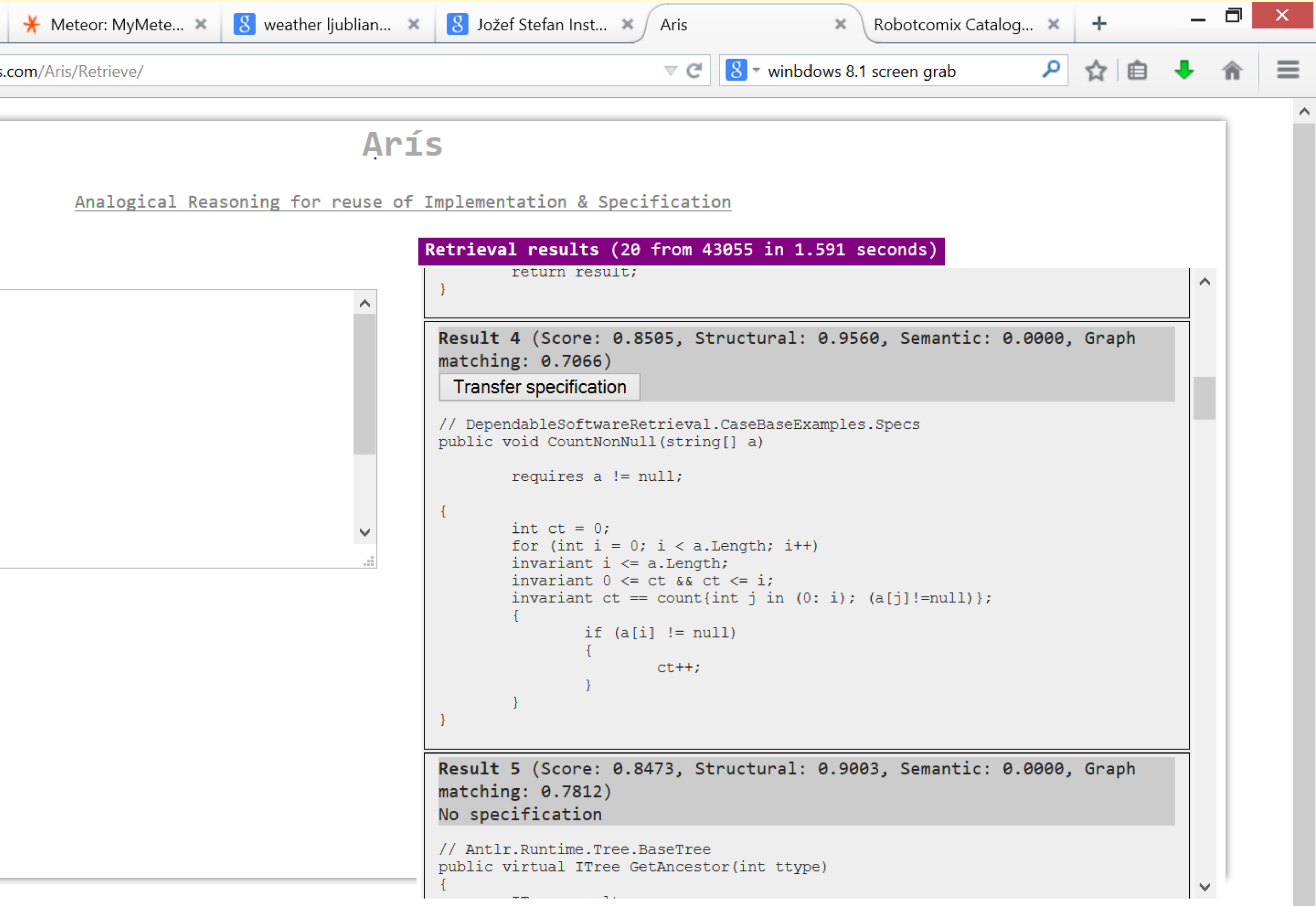
Result 1 (Score: 0.8662, Structural: 1.0000, Semantic: 0.0000, Graph matching: 0.6806)

```
// DependableSoftwareRetrieval.CaseBaseExamples.Specs
public int CountEven(int[] a)

        ensures result == count{int i in (0: a.Length); ((a[i] % 2)== 0)};
        requires a != null;

{

        int s = 0;
        for (int i = 0; i < a.Length; i++)
        invariant s == count{int j in (0: i); ((a[j] % 2)== 0)};
        invariant i <= a.Length;
        {
                if (a[i] % 2 == 0)
                {
                        s++;
                }
        }
        return s;
}
```

Result 2 (Score: 0.8577, Structural: 0.9704, Semantic: 0.0000, Graph matching: 0.7032)
No specification

```
// Microsoft.Exchange.Data.Globalization.CodePageDetect
internal static int GetCodePageCount(uint cumulativeMask)
{
        int num = 1;
        while (cumulativeMask != 0u)
```

O'Donoghue D.P., Pitu M., Grijincu D., Monahan R.
Department of Computer Science, NUI Maynooth. C. Kildare, Ireland.



Arís

Analogical Reasoning for reuse of Implementation & Specification

Retrieval results (20 from 43055 in 1.591 seconds)

```
        return result;
}
```

Result 4 (Score: 0.8505, Structural: 0.9560, Semantic: 0.0000, Graph matching: 0.7066)

```
// DependableSoftwareRetrieval.CaseBaseExamples.Specs
public void CountNonNull(string[] a)

        requires a != null;

{
        int ct = 0;
        for (int i = 0; i < a.Length; i++)
        invariant i <= a.Length;
        invariant 0 <= ct && ct <= i;
        invariant ct == count{int j in (0: i); (a[j]!=null)};
        {
                if (a[i] != null)
                {
                        ct++;
                }
        }
}
```

Result 5 (Score: 0.8473, Structural: 0.9003, Semantic: 0.0000, Graph matching: 0.7812)
No specification

```
// Antlr.Runtime.Tree.BaseTree
public virtual ITree GetAncestor(int ttype)
{
```

Main Technological Innovations

- Information extraction
- Document summarization
- Semantic technologies and ontology
- Model of Analogy & Blending
  - retrieval, mapping, validation *etc*
- Visual analytics
- Evaluation
  - Focused on domain of computer graphics

Conclusion

- Dr Inventor aims to assist researchers
- Finds analogous "documents"
  - With a balance of similarity and difference to a users presented document
- Welcome contact from CC community
  - Sister project called Aris uses "data" in the form of C# source code (& Spec#)