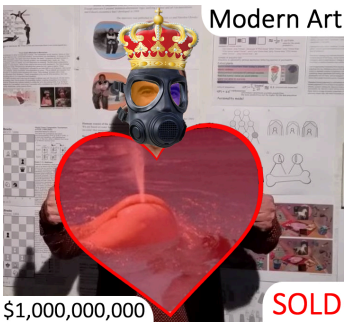
Serverless & Nameless Information System

published on PhilPapers

$$N = \min\left(\prod_{i=1}^{n} C_i\right),\ C_1 + ..C_2 \approx C_n \qquad Q=(n*\bar{q})^{I}$$

Miro Brada



In 2013, I've upgraded the server side each.co.uk system aspx / c# to "serverless" and "nameless" ≈360KB javascript combinable with webassembly without plugins, 3rd party, CSS.. minimizing iterations* per functionality to maximize reuse, with ideas partially presented at conferences in Santorini, Adelaide, Geneva, Daejon and virtually at IS4SI 2021 Summit

- Linguistic principle of arbitrariness of the sign 1916
- Information Theory 1940s
- Multi layer logic of the chess composition 1928 + Model of Inteligence 1998

* iterations fragment functionality. Eg. Achilles can't overtake the slower tortoise: as he moves closer, she moves a bit further, so he moves closer and closer to never reach her. The higher fragmentation in various layers (server, JS, css, sql, frameworks..), the more unproductive interations..

"**Serverless**" runs the all codes in a browser JS, wasm.. except inevitable basic save, send.. data operations run in a few server's commands sql, php, c#, py... The server's functionality is ended, no modifications needed. The old system's outages or users' cut-offs were unresolvable in the server admins installed server's load-balancer to vanish when server's operations moved to the browser. It also absorbed the code in Vanilla **JS**, simplifying the logic. The split code **C** = **S** server **+ B** browser multiplies the cases: **S*B** 2 layers **> S+B** 1 layer. If the code is equally split in server and client: S=B, the code **a** moved from the server: **S-a**, to the browser: **B+a** (B=S), decreases the cases by ≈ $\mathbf{a^2}$: $\mathbf{(S-a)*(S+a) \approx S^2-a^2}$. The ended server's code: **S≈1** prevents bugs on the server, while 2-layers' code S>1, B>1 multiplies the bug's occurence (S*B): $\mathbf{a^2}$. The higher code's concentration in fewer layers, the less cases and the higher reuse reducing the code itself. Usually there are 5 layers: ● server script ● database/SQL ● javascript ● CSS ● html. Number of cases N is: $N = \min\left(\prod_{i=1}^{n} C_i\right)$, where $\mathbf{C_i}$ = cases in layer $\mathbf{n_i}$, and $C_1+C_2+..C_n \approx C$. For $C_1 \approx C_2..\approx C_n \approx C/n$, the code only in 1 layer has n*C/n*1*1.. = C cases, while the code in *n* layers has (C/n)*(C/n)*(C/n).. i.e. $(C/n)^n$ cases. Clearly: $C < (C/n)^n$. Eg. 50 cases in 1 layer is 50, while in 5 layers 10 per layer it's 10^5 = 100,000.

Examples of tiny php files: easy to change to any script: py, pl, c#..

```php
<?php
try{
  include'xxxxx.php';
  function a($x){$p=$_POST[$x];
    if($p)$GLOBALS['P'].= ", @".$x."='".$p."'";}
  $p=$_POST['p'];$P='';a("a");a("b");a("c");a("d");a("e");
  $P=$p.substr($P,1);
  $Q=sqlsrv_query($conn,'exec '.$P);
  $R=[];
  while ($r=sqlsrv_fetch_array($Q,SQLSRV_FETCH_ASSOC))
    $R[]=[$r['a'],$r['b'],$r['c']];
  echo json_encode($R);
  } catch (Exception $e) {}
?>
```

links to SQL (a few php commands)

```php
<?php
try{$n=$_POST['n'];$c=",";$R="";
    file_exists($n)?0:mkdir($n,0777,true);
    $n=$n.'/'.$_POST['p'];
    file_exists($n)?0:mkdir($n,0777,true);
    $a=explode($c,$_POST['a']);
  foreach($a as $v){
    $k=explode("|",$v);
    $f=@file_get_contents($k[0]);
    $r=$n.'/'.$k[1];
    file_put_contents($r,$f);$r=filesize($r);
    $R=$R==''?$r:$R.$c.$r;}
  echo $R;}catch (Exception $e){echo 0;}
?>
```

```php
<?php
if(file_exists($_POST['a']))
  echo file_get_contents($_POST['a']);
else echo '';
?>
```

```php
<?php
file_put_contents($_POST['a'], $_POST['b']);
?>
```

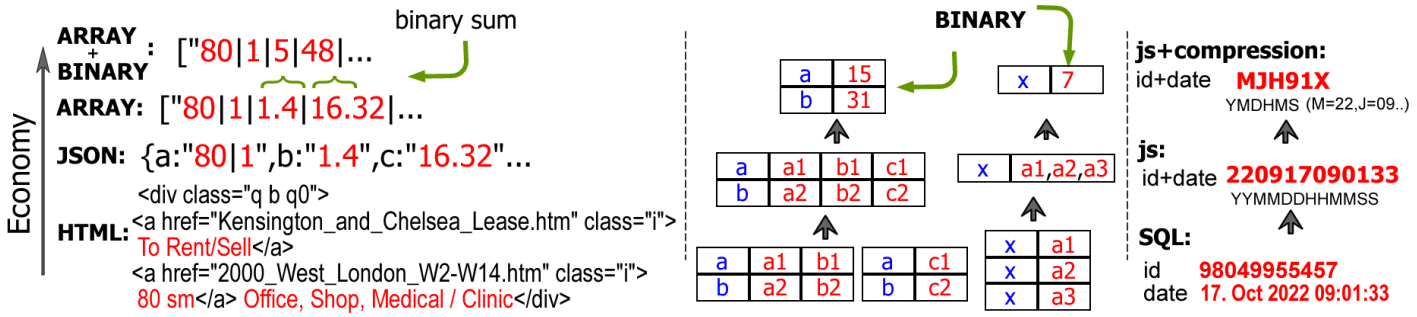
Data and bitwise logic

Ideally data are reused and compressed to minimize the bytes to store and transmit. Data files use the server less than **SQL / DB**, but for the matches or data mix **SQL** is useful. I removed or merged all **SQL** (procedures, meta-data, indexes..). E.g. the javascript date-time replaced the **SQL** identity key. E.g. 17. Oct 2022 09:01:33 ⇒ YYMMDDHHMMSS ⇒ 220917090133 compresible to YMDHMS ⇒ MJH91X (M=22,J=09..). The different users can add the property exactly at the same time, so the properties' ids could collide. To avoid this, the agent's id YMDHMS can be added to the property's id YMDHMS: YMDHMSYMDHMS - 100% unique key in javascript without **SQL** / server.

In the old system, each attribute (property size, property pics, property rent..) had own table to be joined to load the items. E.g. property had 24 sub-tables. It was slowing down the server. I merged the all sub-tables to 1 table, removing the **SQL** keys. Only the main tables remained: Property, Requirement, Negotiator, Branch, Chat, etc. Multiple values

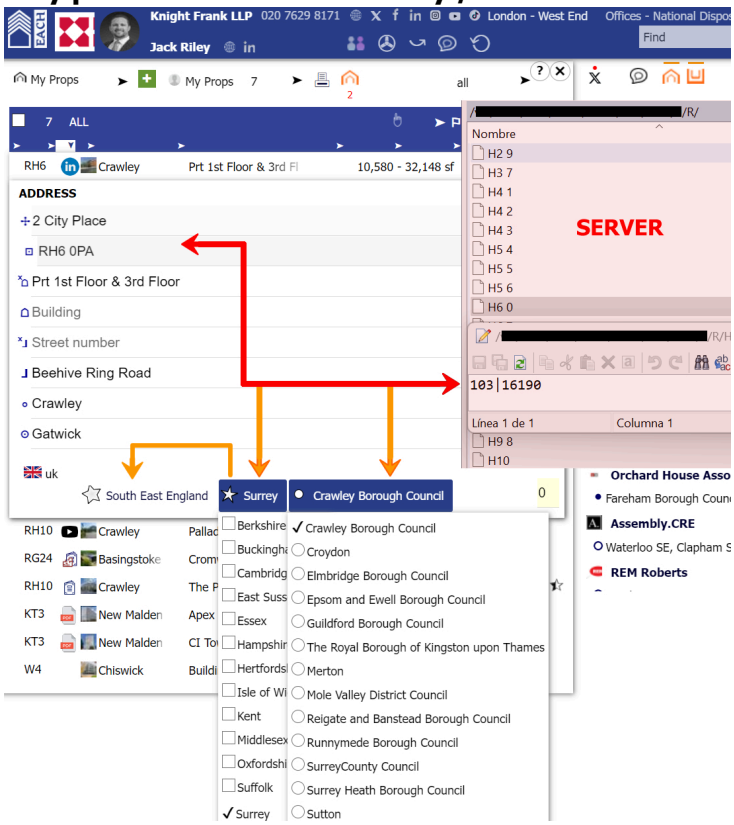
tenures, subtypes, counties.. merged to binary sums or comma separated ids to replace intersection tables. Simple data forms json, array create html in **JS**, replacing the server's generated html. The array seems ideal data form, as the same info in **array**:[5,7,2..] is smaller than in **json**:{"a":5,"b":7,"c":2..} with min 1-letter key (incl apostrophes + colon) adding 4 bytes per key. E.g. 10 properties with 30 attributes load at least (10*30*4) 2.8 KB more from DB if they load jsons instead of arrays.
The bitwise operators can compress and detect multiple values. To utilize the binary sums, every value must follow series: 2^x, $x \in \{0, 1, 2.., n\}$. So the values of attribute can be: 1, 2, 4, 8, 16 etc.. Eg. for tenures: lease=1, short lease=2, freehold=4, long lease=8. Then if the property is lease (=1) and freehold (4), its tenure is (1+4) 5. Or the requirement for freehold (4) and long lease (8), has tenure 12. Then the bitwise "&" detects if eg. Property's tenure (X) matches the Requirement's tenure (Y): X&Y>0. In our example 5&12=4 - they match. This is much faster than detecting the match in strings '1,4' vs '4,8'. **SQL** server limits binary to 2^31-1 B, so there can be max 30 multiple values in one column. In our database there are over 100 subtypes, so we have 3 columns of binary for Properties' subtypes: X_1, X_2, X_3, and 3 columns for Requirements' subytpes: Y_1, Y_2, Y_3.. And the matches are determined by $X_1\&Y_1>0$ or $X_2\&Y_2>0$ or $X_3\&Y_3>0$. The binary value is calculated in javascript when property / requirement is created or edited. Also because we keep the order of displayed subtypes we still store comma separated subytpes' ids.

Data form compression

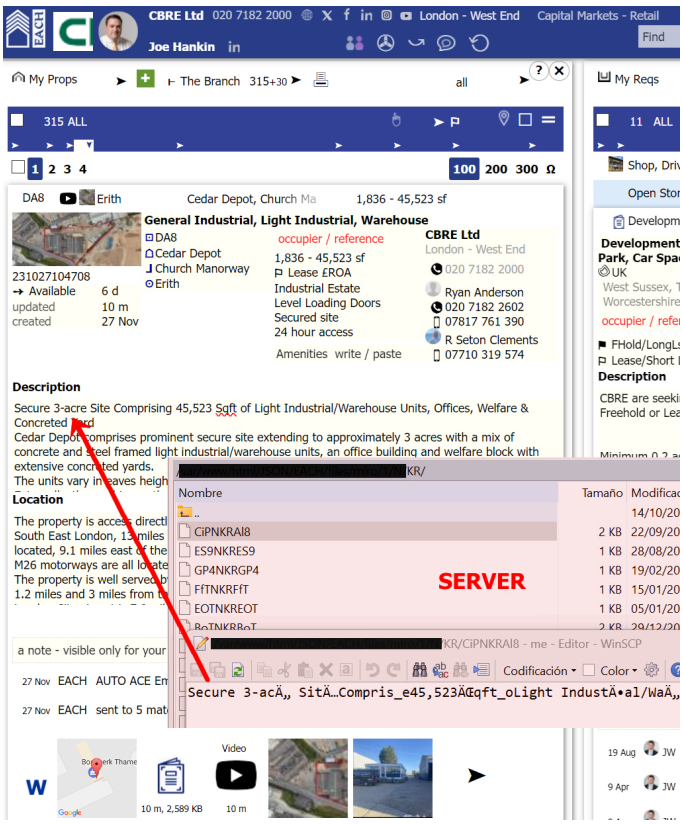


The long texts as descriptions are compressed using keywords replacements or lossless, Huffman algorithm in **JS** before saving. Js also decompresses the file, loaded only if needed: eg. the description is loaded only as the line/grid view is clicked to show detail. If the text is needed in the keywords' search, the texts can be saved also in the reduced but uncompressed form: without repeated words, in the **SQL** used only for the keyword's search - the text itself always loads from the compressed file. All replaceable **SQL** was removed. Eg. as the Property was created, its county id was assigned (based on postcode) from the 4 MB **SQL** table of 32K rows, which slowed down the server. I replaced it by a simple 52 KB files' system with 4 postcodes' letters to get the county / council id by **JS**.

Tiny postcode's file with county / council id



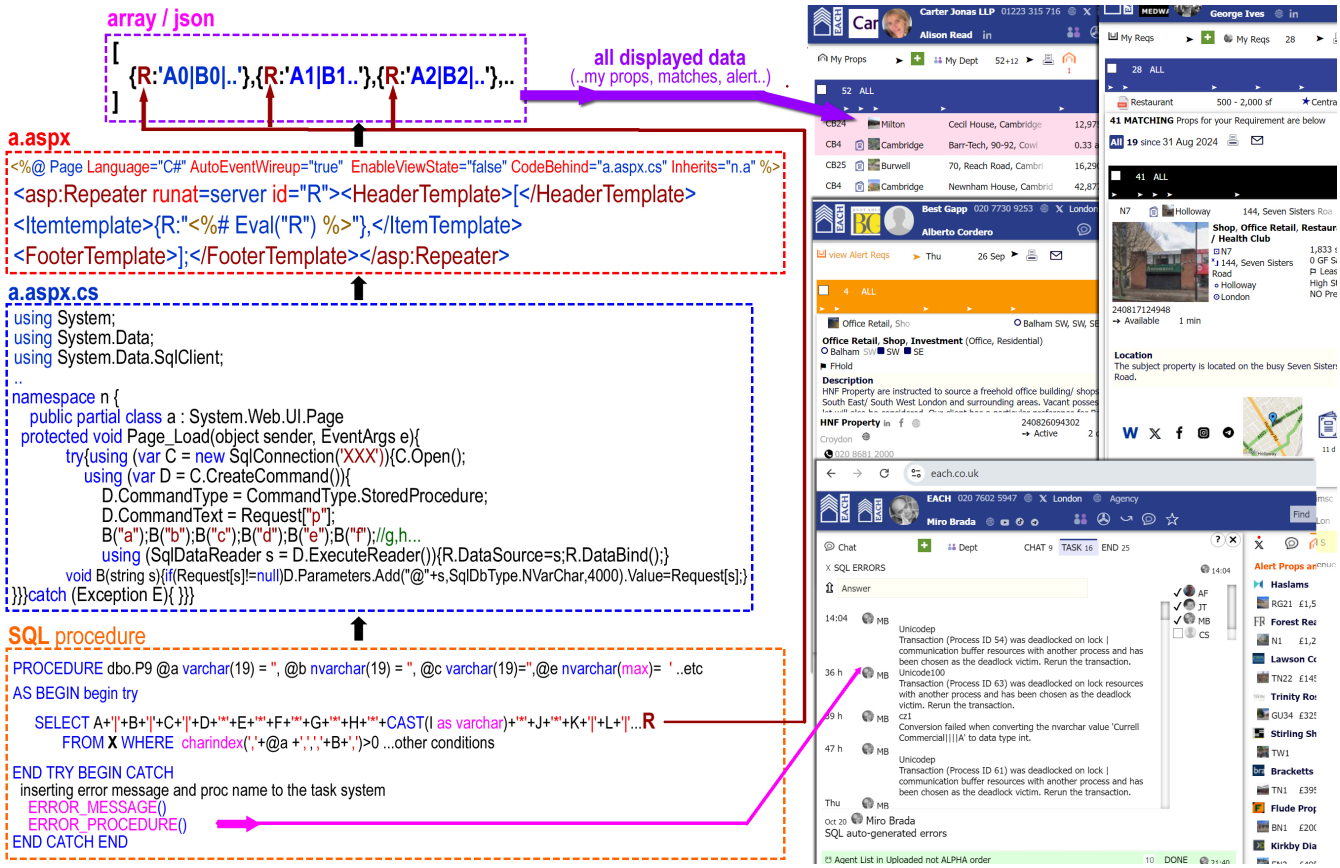
Text from the compressed file in the detail view



Uniform data form & ajax / fetch

In 2002-2013, each.co.uk used over 30 aspxs' pages: properties.aspx, requirements.aspx, invoices.aspx, department.aspx.. with specific **SQL** procedures. The url had to reload to view the pages keeping server's user session. AJAX (asynchronous javascript + XML) arose in 1999 (fetch in 2015), enabling single-page app (SPA) to load data without re-loading the url. Using ajax, I merged the all aspxs to one aspx loading single parameter 'R' uniting the 'SELECT' of all **SQL** procs with uniform data form:[{R:'A0|B0|'},{R:'A1|B1|'},..]. I reduced the aspx / c# code to tiny C# scripts. In 2019 I rewrote them to php in 2 days. And I replaced jsons by arrays: [['A0|B0..'],['A1|B1..'],..] to reduce the size of loaded data.

SQL ⇒ C# ⇒ aspx ⇒ json mechanism to load all data



Security and deployment

"Serverless" system without 3rd party programs and plugins, has fewer options to be attacked. The only vulnerability resides in running **JS** in Inspect Element / console. To obfuscate the code is helpful, but not an obstacle for an experienced hacker. Also deleting the header's references is right, so if the page is downloaded, there is no **JS** file. But the IE console shows the **JS** functions as 1st letter is typed - so the functions from the **JS** deleted files can be revealed. The main protection seems to disable IE via keycodes / right mouse click, and replace the codes as the IE console opens. Further, to insert / update data, the hash from the server is required. Parts of the code can be in webassembly whose compiled binary hides the code. This is just a brief explanation without details.

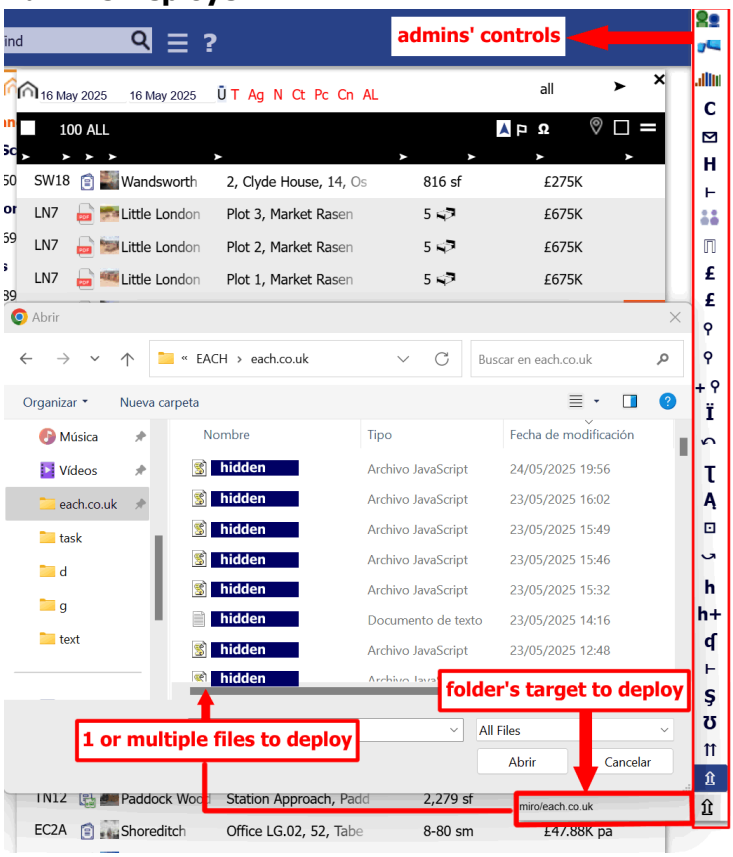
The old system used *MS Visual Studio* for development with *source safe*. I removed it to develop a tiny ⇫ **deployer** as Admin control to open and select files to be deployed at the folder specified in the input box the last value is default. Mostly **JS** files are deployed, but any file including server's php, images or wasm can be deployed. It also auto-backs up the files replaced by the new ones. ⇈ **Uploaded stats** shows the deployed and old files, size change, who and when deployed, and option to view the files. For a team-work, every programmer has own **JS** file to write his code (functionality). To define the global function / variable - Admin's ∅ **uniquer** shows a free 2-letter name - unused by any global function / variable.

3rd party programs, plugins, frameworks increase vulnerability. No surprise, there've been regular and serious outages of renowned companies eg. TSB bank, Barclays, British Airways, Mark & Spencers.. A brief view of their websites' sources, reveals many codes & plugins on their front-pages (after search / login more files are loaded). The foreign codes are uncontrollable and so decrease security, performance and compatibility.

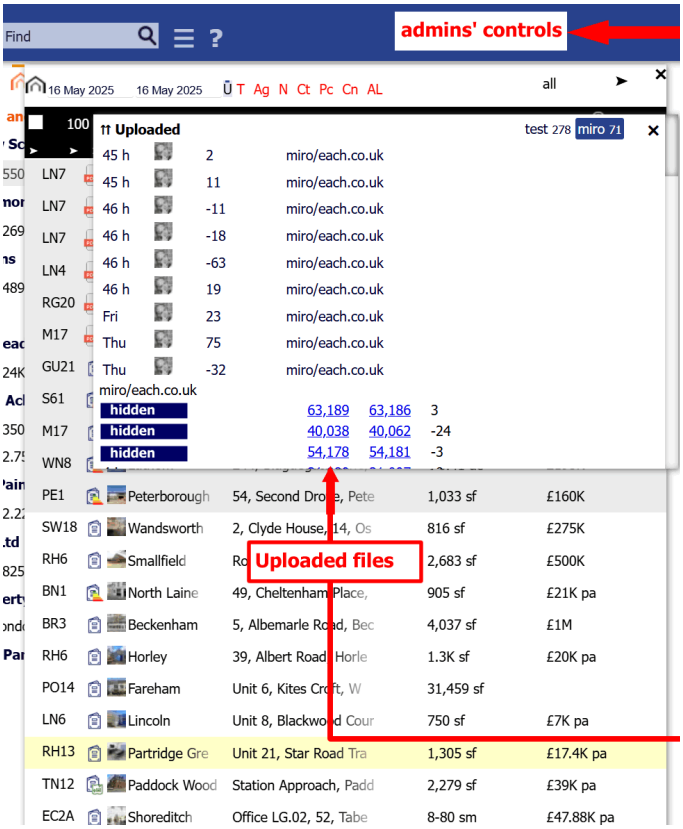
Except each.co.uk I developed the public site 4prop.com to capture enquries from searches or SEO pages. The emails enquiries have gradually in 2 years increased from 0 to 45 a day. Each.co.uk's deployer was deploying the 4prop's files and other companies' websites with each.co.uk's **JS** search. In Sep 2014, a new programmer against my advices and will, replaced 4prop.com by codes using many plugins. It destroyed the SEO enquiries, and increased the dependency on the 3rd party. The admin in Jun 2020 installed 'docker' unneeded for each.co.uk, but

needed by new 4prop to deploy the codes. Each.co.uk and 4prop.com used separate DB (each.co.uk DB was copied at night to 4prop DB), but both had access to the files. On Friday 28 Aug 2020, the hacker via 'docker' started encrypting the attachments and pics. Marcel - other programmer, noticed it on Sat night; almost all files had been ecnrypted. I stopped the server. On Mon our admin retrieved the backed up files before the attack. Mon was bank holiday - if unnoticed, the backed up files could be encrypted too. So we almost lost data, only because of wrong decision I couldn't affect.

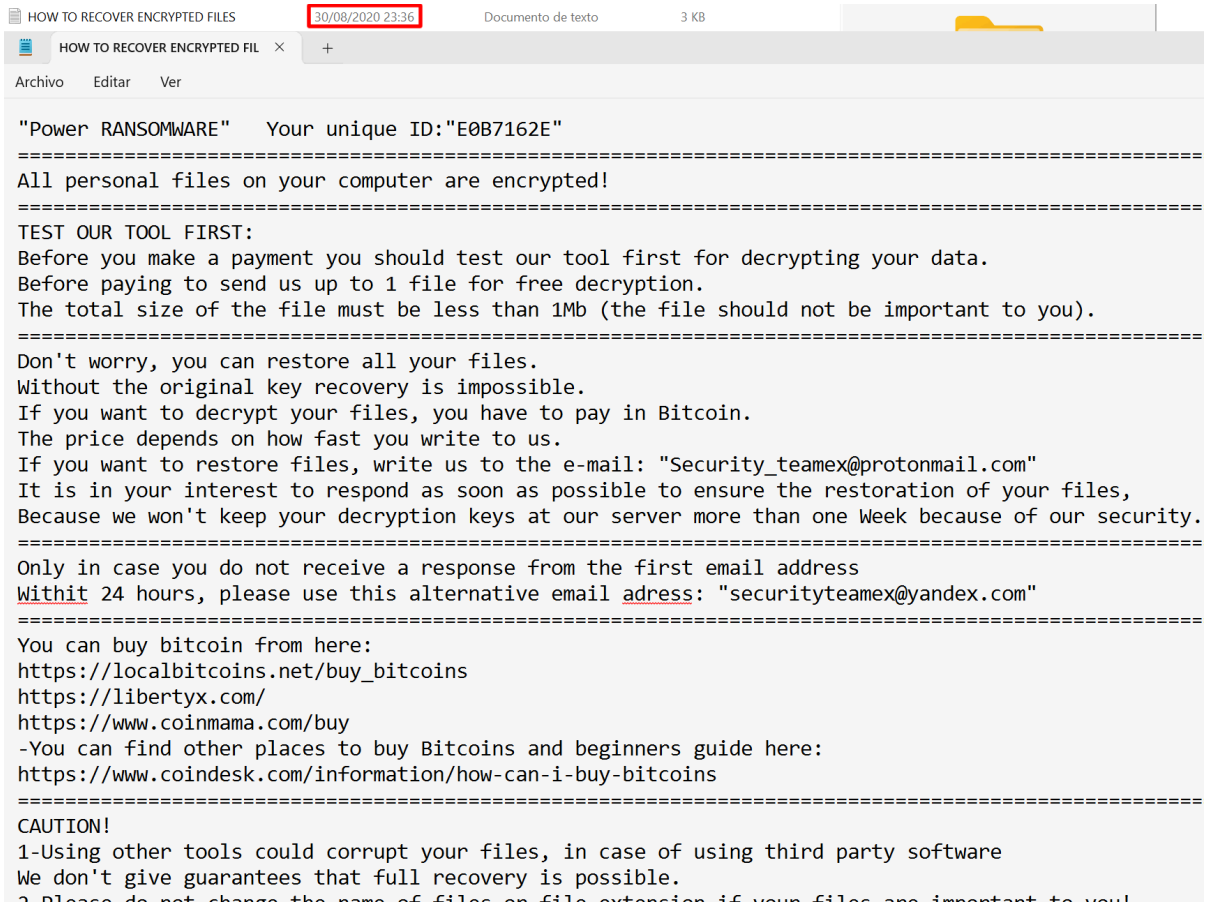
Admin's Deployer



Admin's Deployer's statistics



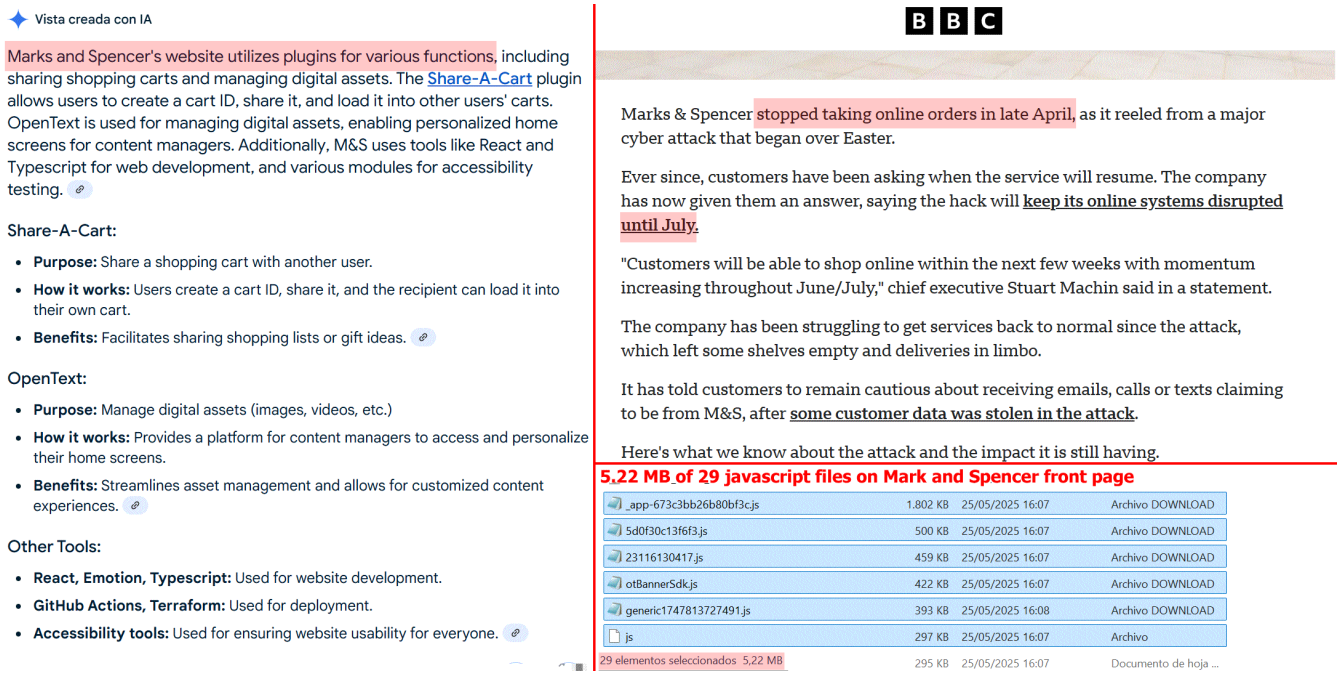
Ransomware attack on Fri 28. Aug 2020, from the 3rd party "Docker" used by 4prop.com (not each.co.uk)



```
"Power RANSOMWARE"   Your unique ID:"E0B7162E"
================================================================================
All personal files on your computer are encrypted!
================================================================================
TEST OUR TOOL FIRST:
Before you make a payment you should test our tool first for decrypting your data.
Before paying to send us up to 1 file for free decryption.
The total size of the file must be less than 1Mb (the file should not be important to you).
================================================================================
Don't worry, you can restore all your files.
Without the original key recovery is impossible.
If you want to decrypt your files, you have to pay in Bitcoin.
The price depends on how fast you write to us.
If you want to restore files, write us to the e-mail: "Security_teamex@protonmail.com"
It is in your interest to respond as soon as possible to ensure the restoration of your files,
Because we won't keep your decryption keys at our server more than one Week because of our security.
================================================================================
Only in case you do not receive a response from the first email address
Withit 24 hours, please use this alternative email adress: "securityteamex@yandex.com"
================================================================================
You can buy bitcoin from here:
https://localbitcoins.net/buy_bitcoins
https://libertyx.com/
https://www.coinmama.com/buy
-You can find other places to buy Bitcoins and beginners guide here:
https://www.coindesk.com/information/how-can-i-buy-bitcoins
================================================================================
CAUTION!
1-Using other tools could corrupt your files, in case of using third party software
We don't give guarantees that full recovery is possible.
```

On 21 May 2025, BBC informed that Mark & Spencers hasn't taken order since end of April and the disruption will be until July! Downloading the M&S's website shows 5.22 MB of 29 **JS** files only on their front page, while the biggest **JS** file has 1.8 MB! The AI names long lists of various plugins or programms utilized by M&S website..

Mark and Spencer 21 May 2025. The front page: 5.25 MB's **JS** with many foreign **JS** limiting the reuse & security



Vista creada con IA

Marks and Spencer's website utilizes plugins for various functions, including sharing shopping carts and managing digital assets. The Share-A-Cart plugin allows users to create a cart ID, share it, and load it into other users' carts. OpenText is used for managing digital assets, enabling personalized home screens for content managers. Additionally, M&S uses tools like React and Typescript for web development, and various modules for accessibility testing.

Share-A-Cart:

- **Purpose:** Share a shopping cart with another user.
- **How it works:** Users create a cart ID, share it, and the recipient can load it into their own cart.
- **Benefits:** Facilitates sharing shopping lists or gift ideas.

OpenText:

- **Purpose:** Manage digital assets (images, videos, etc.)
- **How it works:** Provides a platform for content managers to access and personalize their home screens.
- **Benefits:** Streamlines asset management and allows for customized content experiences.

Other Tools:

- **React, Emotion, Typescript:** Used for website development.
- **GitHub Actions, Terraform:** Used for deployment.
- **Accessibility tools:** Used for ensuring website usability for everyone.

BBC

Marks & Spencer stopped taking online orders in late April, as it reeled from a major cyber attack that began over Easter.

Ever since, customers have been asking when the service will resume. The company has now given them an answer, saying the hack will keep its online systems disrupted until July.

"Customers will be able to shop online within the next few weeks with momentum increasing throughout June/July," chief executive Stuart Machin said in a statement.

The company has been struggling to get services back to normal since the attack, which left some shelves empty and deliveries in limbo.

It has told customers to remain cautious about receiving emails, calls or texts claiming to be from M&S, after some customer data was stolen in the attack.

Here's what we know about the attack and the impact it is still having.

5,22 MB of 29 javascript files on Mark and Spencer front page

_app-673c3bb26b80bf3c.js	1.802 KB	25/05/2025 16:07	Archivo DOWNLOAD
5d0f30c13f6f3.js	500 KB	25/05/2025 16:07	Archivo DOWNLOAD
23116130417.js	459 KB	25/05/2025 16:07	Archivo DOWNLOAD
otBannerSdk.js	422 KB	25/05/2025 16:07	Archivo DOWNLOAD
generic1747813727491.js	393 KB	25/05/2025 16:08	Archivo DOWNLOAD
js	297 KB	25/05/2025 16:07	Archivo
29 elementos seleccionados 5,22 MB	295 KB	25/05/2025 16:07	Documento de hoja ...

"Nameless" Information

"Nameless" merges the layers CSS, html, sql with own names to single layer JS to reduce the names definitions running the system. In 2017, I presented my lecture Psychological and other aspects of the sign arbitrariness at the conference in Geneva *The Arbitrariness of the Sign*, an idea (1916) of linguist F. Saussure: the word signifier and its representation what signifies is arbitrary.

Names' Reuse

Complex logic eg. language, reuses its units. European languages have c. 30 letters whose combinations create all meanings. Abstract characters: "a", "b".. are easily reusable, unlike chinese pictographs: 2-3000 needed to read the newspaper. Because non-reusable characters disable grammar, the chinese has 'radicals': reuseable 'arbitrary' units amending the meaning of the pictograms. Eg. a 'radical' denoting a person: **人** rén or its shorter form: **亻** is added to other characters. Everyone: **人人** rénrén, estimate: **估** gū, imitate: **仿** fǎng, night: **夜** yè.

- The more abstract units, the higher reuse.
- The smaller reuse, the harder to construct a logic. Chinese characters even with radicals aren't abstract enough for advanced grammar. It's very simple: no tense, verbs don't change.. Except 'radicals' the chinese uses 'measure' words adding extra specifications to compensate simple grammar. They are added before nouns: 个 gè is more general, and other more specific as: 只 zhī to count animals or single items in pairs, 张 zhāng for flat things: table, paper, pic.., 碗 wǎn for bowls..
- The simpler grammar, the higher divergence of logic. The chinese-similar languages Mandarine, Cantonese, Japanese, Koreans, Vietnamese.. differ a lot more than european languages eg. Slavic Slovak, Polish, Russian.. or Latin Spanish, Italian, Romanian... As "Deaf and dumb language" sign language with 'shown signs' uneasy to reuse, isn't universal, with own distinct versions across countries / regions.

Names' reduction

Long names prolong the code, while real names as '*save*', '*property*', '*date*'.. creates an illusion of understanding. The real name is as one-use 'pictogram' without variations. Eg. chinese 'home' 家 jia is 'pig under roof', but it could be: 'cow under roof', 'head under roof'.. and/or 'pig in a room', 'pig next window'.. Or if 'pig under roof' is home, what's 'pig under tree'? A cottage? No, a cottage is 小屋 xiǎowū - small house.. This shows countless variants for the same thing. The real name pictogram can be a perfect hint, but in more contexts it confuses / misleads.
In 2018, I applied 1-2 letters' abstract names for functions, variables, tables, columns, SQL procs. I did a function to find free 2 letters for unique name undefined in JS. The set for the 1st letter are 26 [a-z] letters or [A-Z] capitals or underscore _: 26+26+1=53. The 2nd letter can include 10 digits [0-9], i.e: 53+10 = 63 possible signs. It's: 53*63=3,339 unique 2-letters' names. On 1 June 2025 each.co.uk used 2,546 76% names for complex functionality emails, chat, invoices, views, users, properties, requirments, companies, alert, filters, archive, searches, import, prints, ETL, SEO etc. The

names repeat at least 2x, but often several times. If 2,564 names would have on average 3-letters, it's: 1*2,564*2=5.128 KB extra. For 4-letters repeating 4x, it's 2*2,564*4=20.512 KB extra. The code was reduced by 50+ KB, as the names often exceeded 4 letters and 4 iterations.

Shortening the names doesn't reduce the names. Their reduction results from moving **CSS**, **HTML** and **SQL** structures to **JS**. In 1994, *Håkon Wium Lie* introduced **CSS** to define the styles. It splits the code needing an extra name: class or attribute, while **CSS** is definable "namelessly" in **JS** whose names style's constants can be endlessly reused. So for the same style, **JS** needs far less names and bytes of code than **CSS**.
To illustrate: **CSS** can name widths as [***bX***]{width:**Xpx**}.. [***b0***]{width:**0px**}[***b10***]{width:**10px**}[***b35***]{width:**35px**}.. Each "**Xpx**" width has own name "***bX***". In contrast, mere 4 **JS** names constants eg. ***Sy***, ***fh***, ***pX***, ***wQ*** can define any width: *const* ***Sy***=' **style="**' ,***fh***='**width:**',***pX***='**px"**' ,***wQ***=***Sy***+***fh***. Then: **CSS** [***b0***]{width:**0px**}[***b10***]{width:**10px**} is equivalent to **JS** ***wQ***+**0**+***pX***, ***wQ***+**10**+***pX***..
CSS name is longer and each value has own name. For {**0,1,..,9**}**px** widths: **CSS** name has 15 letters - while **JS** 7, for {**10,11,..,99**}**px** widths: **CSS** name has 17 letters - **JS** 8.. For 100 unique {**0,1..,99**}**px** widths: **CSS** has 100 names and 1,682 KB 10*15+89*17+1*19 B, and **JS** has 4 names and 0.836 KB 10*7+89*8+1*9+45 B (for JS names), so **JS** has 96% less names and 50.3% smaller code.. The more code, the bigger difference.

10 June 2025, **491 KB CSS** file on British Airways's website

CSS styles height, color, background, opacity, padding, position, radius.. are often variously combined in **CSS** attribute or class, further expanding the names, while **JS** combining styles's names namelessly. **CSS** can define just one name for each attribute color height with.. to combine names in **HTML** with multiple names eg: div element '**<div** class="***ab cd ef***"**>**' links 3 classes: .***ab***, .***cd***, .***ef*** instead of one class combining styles. One attribute and value per **CSS** name increases name's reuse, but in practice the multiple styles in **CSS** name is common. One of the British Airways's **CSS** file has eg. 16 letters' *background-color* attribute, often combined with other attributes, in 111 classes. If *background-color* is defined as 2 letters' **JS** constant, it would save 14*111=1.554 KB.
Replacing multiple styles would further reduce over 110 names and more bytes - just for one **CSS** name. Reducing (by 2 letters' **JS** constants) 4 other attributes eg. *line-height* 9x82=738, *font-size* 7x64=660, *padding-bottom* 12x55=660, *margin-bottom* 13x47=611 would remove 2.669 KB. So moving merely 5 **CSS** names to **JS** would reduce 4.233 KB and 354 names. **CSS** values as *1000px*, *transparent*, *sans-serif*, *relative*.. can be also defined as **JS** constants to further reduce bytes.. Eg. in British Airways' **CSS**, there is 6x 37 letters' value *rgb(52 104 173/var(--tw-bg-opacity,1)* = 222 B (in 2 letters' **JS** 12 B). Or 28x 11 letters' *transparent* = 308 B (in 2 letters' **JS** 56 B).. Defining style's values as **JS** names reduce the bytes but increase the number of names. So it's rational to use **JS** names for style's values, if they repeat many times.

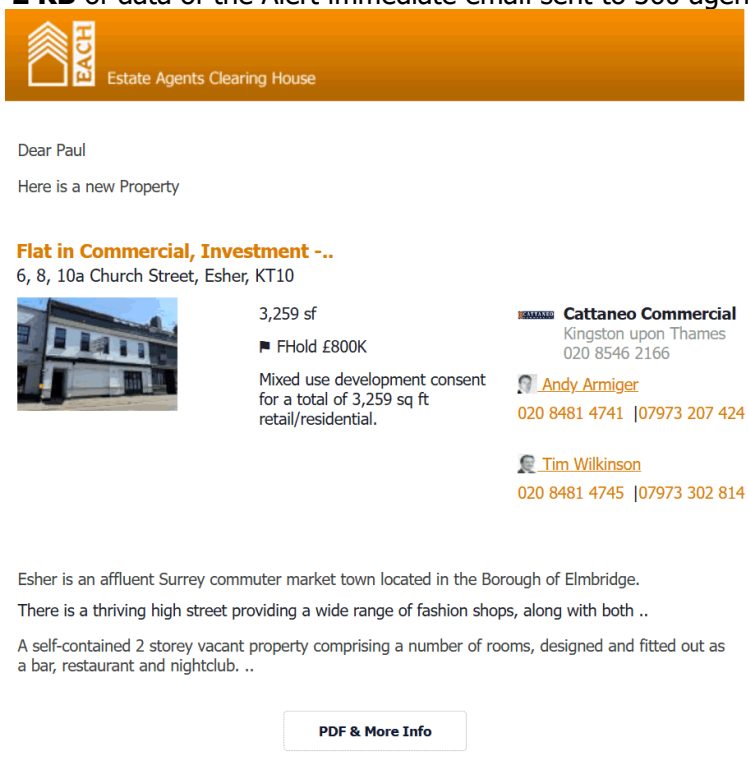
HTML app's skeleton used to be in the files templates filled by data on the server. Since ajax / fetch enabled to load data asynchronously, the html has been more often created in the browser, although the html files are still common to create emails, SEOs, reports pdf, doc... But, **HTML** is also definable by **JS** names, eg: *const* ***_i***='**<div** ' ,***_P***='**<span** ',***D1***='** **' ,***s8***='****'
HTML in the file with full style: ** **
Or with the style in the separated **CSS** files: ** ** + .***b10***{width:**10px**}
Or: **<div b10> ** + [***b10***]{width:**10px**}

By **JS** names, it's the incomparable shortest: ***m_***+**10**+***Yj***, where: ***m_***=***_i***+***wQ***,***P8***='**>**',***y9***=***pX***+***P8***,***Yj***=***y9***+***D1***

The combined **JS** names create any **HTML** of any style for website, emails, SEO, reports.. to hugely reduce the structure / code / cases.. **JS** auto-job runs in the browser (as EACH Admin logs in), fetching files / DB's data, to make emails, import / export / checks.. 7 Admins used to login - sufficient to run the **JS** auto-jobs. Later I added the auto-jobs to be run by all users' sessions - a tiny (1%) fraction of random users. **JS** auto-jobs aren't intensive and can't cause any browser's outage. "Just-in-time" control file flag prevents running the same auto-job mutiple times in different browsers.
On the server, there is mere 1 job: a tiny php file sending 1 simple loop (no data processing) the emails whose html / data were created in **JS**. Emails are created / sent: 1) as the item property, requirement.. is created / edited, 2) by user's action ACE / Message/Enquiry, 3) by **JS** auto-job.
Eg. *daily Alert* emails items n_1, n_2..max 3 properties and 3 requirements per email to M users: N*M, while the same item n_x

is sent to multiple users. The old system called c. 7 SQL functions to get: address / location, types, size, tenure, amenities, descriptions, agents for every single n_x item to be created, while the Nameless system creates the n_x only in **JS** with no SQL call. Each.co.uk sends c. **40K+** various emails a day: 2x ACE: Prop / Req, 2x Matches: Prop / Req, 2x Alert immediate: Prop / Req, 2x Register Check: Prop / Req, Alert daily/weekly, Tasks / Chats, Enquiries. C. 5-10K are *daily Alert email* - we can assume 7.5K with 3 items of max 6 on average per email: 7.5K*3*7 = 157.5K SQL calls, in contrast with 0 calls by the nameless system. Other emails have 1 item per email, except Matches that can contain more items. Overall, we can assume 1.5 item per email: 40K*1.5*7=420K SQL calls per day to create emails by the old system. The SQL calls queried 1 or more joined tables - as the old system had fragmented DB's structure. And the Matches and Alert matches used the intensive string's id searches instead of binary sums. So regular SQL / server outages occured.

2 KB of data of the Alert immediate email sent to 360 agents



Except reducing the server's **SQL** overload, **JS** reuses itself to remove the duplicated **SQL** functions that sharply increase maintenance costs and probability of bugs, as modifications or new functionality had to be in **JS** as well as **SQL**, and in the old system in C#/ASPX too. One programmer used to re-compile **SQL** for emails / reports to mimic the **JS** change. The emails or htm pages (as SEO) can use template with **css** styles (instead of longer nameless inline style) to reduce the size of email sent from the server. Unlike web browser with the constructed style from the 2-letters **JS** names, the emails' browser cannot make their style dynamically from **JS** names. So althouth the emails are namelessly created by **JS**, the html email must contain the full style: ** **, not nameless style: ***m_*+10+*Yj***. Here it's shorter to use html template with css. But the emails' browser as *gmail*, *yahoo*, *outlook*.. differ from each other more than the web browsers. Especially *outlook* can show css differently. And so a bit bigger emails' size using inline styles can be compensated by the more consistent look in the different emails' browsers. Overall, it's better to minimize the emails' html structure and inline styles to create emails namelessly. Or to slighly enhance **JS** functions to use css's classes for emails / htm pages.

To sum up, **JS** names universally increase the reuse of the code, to radically reduce the costs of development and (server's) performance. Another option to dynamically create HTML, style and functionality is **JS** DOM Document Object Model, which is however more complicated and less reusable with more specific commands / names. And except extra load for the browser, DOM commands can become obsolete or incompatible in different browsers.

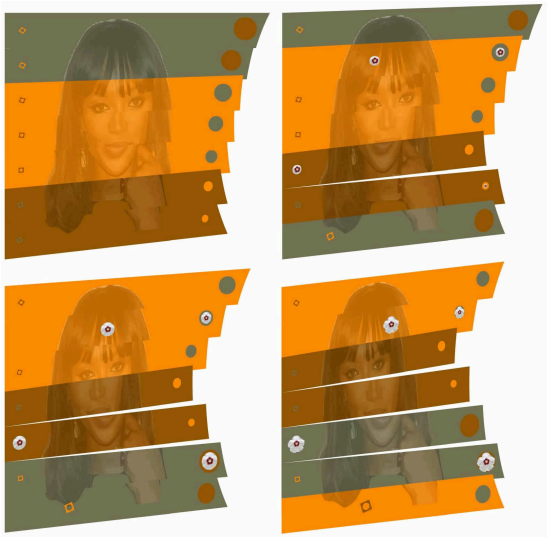
Plugins "imitate" calculus

Since 2000, various **JS** plugins libraries / tools / frameworks have appeared to hasten the development. **WordPress** 2003 Matt Mullenweg, Mike Little, **Dojo** 2004 Alex Russell, **jQuery** 2006 John Resig, **node.JS** 2009 Ryan Dahl, **Bootstrap** (Twitter) 2010 Mark Otto, Jacob Thornton, **Angular** (Google) 2010 Miško Hevery , **React** (Facebook) 2013 Jordan Walke , **Vue.JS** 2014 Evan You.. Many sometimes contradictory views claim why to use or avoid the plugins, or which one is better.. The plugin is a meta or super language with extra namespace to manage usually **JS** commands. It increases the size, reduces flexibility and speed, can corrupt security, needs to be updated.. The advantage is pre-built functionality or auto-adjustments for mobile, PC.. It seems powerful and cool as to say 'be light' to be light.. But unlike calculus it doesn't bring the new quality, because the plain **JS** can do all what plugins do. Calculus is more than the set of calculations, it sums and combines the lower calculations to capture the infinite series to exactly calculate eg. the globe's space. Most likely there is only one way to define calculus, invented independently by Leibniz 1673 and *Newton* 1665. None of the plugins defines the universal logic to combine commands to the new quality. Moreover, the plugins aren't to be understood, but to "free" programmers from the "lower calculations" to focus on the functionality. But can be functionality well developed without understanding the underlying algorithms? Seemingly, it's as surgeons don't need to understand their tools microscope, scalpel.. to operate. But the surgeons can't manufacture their tools themselves, while programmers can replace any plugin by **JS**, except specific features eg. google map plugin with huge data of images, positions. For programmers, the surgeons' tools are hardware (PC, laptop) not plugins that are only a method of work. So, the plugin itself isn't such a big "magic wand" as it looks, while it's "a pig in a poke" without a proper control.

"1-layer" systems

Corporations at the West, in China / elsewhere can be different own many or most programming jobs, with a specific motivation to use multi-layers frameworks / plugins prioritizing control and monopolization over efficiency. As a result, the job-market often prefers the plugins' experience. It forms programmers' motivation and opinions leading to 'fan-clubs' of endless self-confirming views. It creates a self-fulfilling prophecy: the plugins / frameworks seem inevitable to standardize functionality with a support, community, knowledge-base. It seemingly lowers the dependency on the individual skills, so the programmers are easier replaceable to reduce the company's risk. But in practice it's not so stunning, as mentioned failures of IT frameworks' systems: *Mark & Spencer*, *BA*, *TSB*.. Plugins' 'fun-clubs' would say: it would happen or be worse without plugins.. Also it's uneasy for companies to find programmers to build up the pluginless system, which can be intention of the IT corporations to prevent competition. To ditch the framework means to transfer the functionality that's often too expensive.

From animation, Holland Park, 2013
4 images of the series created in Flash Action Script



But **Netflix** in 2017 replaced the **React** by **JS** in login & landing pages to significantly (50%) improve the performace. Or in 2024 **Microsfot** abandoned **React** for its **Edge** browser development, due to inefficiency.. Moreover, the plugin can become obsolete. **Flash Adobe Player** (1996) enabled advanced including interactive animations in its Visual Studio / Action Script. **Flash**'s **fla** compiled to **swf** files were running in browsers' plugin. In 2020, it stopped being supported due to a security vulnerability. In 2003 I started using **Flash** for animations and app selling perfumes. At that time, **JS** couldn't create such visual effects. Even today the **HTML5** with **canvas** element (2005) can't do all what **Flash** could. My acquaintance co-working on **Flash** projects, claimed the **Flash** is the best as it works in the all browsers, and those not using it are 'stupid'.. Earlier, the differences among the browsers were bigger than today (2025), and the **Flash** visuals were far superior to **JS**. Later, **JS** has enabled advanced animations too, so **Flash** became less exclusive with unresolved security, resulting in its end, although **Flash**'s swf can still run in **Ruffle** Mike Welsh, 2016 as browser's extension.

In 2013, I applied **Flash** for map's display and drawing area on each.co.uk, but shorlty after I replaced it by **JS**. Then, I was using **Flash** only for animations presented at my art exhibition "From Animation" in Holland Park Oct 2013. For 3D effects I used **Blender** Ton Roosendaal, 1994 with **Python** Guido van Rossum, 1980s, and **Videopad** NCH Software, 2008. The similar idea to **Flash** is **Java** James Gosling, 1991 with **JVM** Java Virtual Machine running on all server's platforms. Flexibility is at the cost of lower performance / higher memory consumption. **Java** has become faster, but its popularity has been falling, namely since 2020. Unlike **Flash** focused on presentation / games no data, **Java** hasn't depreciated as many data apps (banks, telco..) use it. So the language can survive not because it's efficient, but because it creates too many dependencies costly to replace. Different example is **COBOL** Grace Hopper started in 1959, still used by 43% of banking systems Reuters, 2017: powering 65+% of enterprise software and 70% of business transaction processing, incl 95% of ATM swipes. It's not used for the new programs, but it persists not only because it's costly to replace as **Java**, but due to its operational efficiency with *"superior stability and processing power"*. As Google's AI states, a motif to create **COBOL**: *"the need for a universal business-oriented programming language to run on different computer systems, reduce development costs, and speed up business operations"*. In fact, **COBOL** runs in **1-layer** - a crucial attribute of *serverless* system. To run on "server" or "browser" is irrelavant here, as data operations can't be always run in the browser. The key thing is that the code isn't split in 2 or more layers. Also the older languages as **Fortran** John Backus 1953, **COBOL**, **C** Dennis Ritchie, 1972, **C++** Bjarne Stroustrup, 1979 had arisen before the internet (or ajax/fetch).. They are efficient as they operate in **1-layer**. And, at those times, the efficiency was necessary as the computers were far slower / inflexible, so the programs to work, had to be very economical. Cheaper and faster computers reduced the necessity of 'super' efficiency especially for web apps, while commercialization and outsourcing since 1990s put the profit before the efficiency. While the inefficient programs can suffice initially, as the complexity of processing raises, the maintaining costs hugely rise with potentially unresolvable issues costs, performance, security or other.

To sum up, the older systems as **Unix** Ken Thompson, Dennis Ritchie, 1969 operate super efficiently up to date as they arose from different motivations / environment than commercial apps as **Facebook** Mark Zuckerberg, 2003, **Amazon** Jeff Bezos, 1994 etc.. There are some exceptions as **Telegram** Pavel & Nikolai Durov, 2013, with the same functionality as Meta/Facebook, but with far less resources mere 304 employees, in contrast with Meta's 75K (2025). Or *open-source* Chinese AI **DeepSeek** 2023, Liang Wenfeng performed on par with *proprietary* American **OpenAI** 2015, for a fraction of a cost, resulting in Jan 2025, to the historical fall of the US stock market chipmaker Nvidia: -18%. The increasing proprietarization and monopolization controled by the 'market' - not quality control, oversimplifies IT products' assessment creating biases and overexpectations such as a dot-com bubble in March 2000. This mental outsourcing

avoids understanding of the intricate ideas, while still decides about their "market" value, to lower chances to recognize / promote the best ideas.

Quality of Algorithm

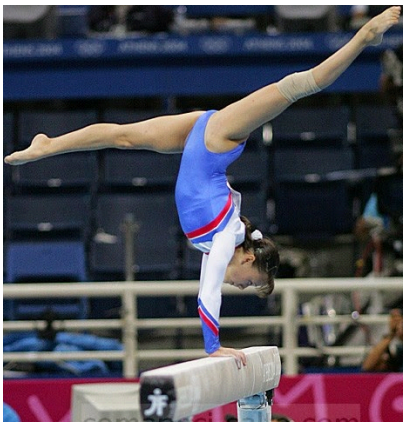
Descartes's La Géométrie, 1637 with XY depiction of math, means: any visualization incl. app is definable by one logic. *Newton* in his Philosophiæ Naturalis Principia Mathematica, 1687 elaborated the Descartes's idea unifying math and reality its visualization to include his invention of calculus 1965. So the new quality Newton's laws roots in the unifying view of realities. And the apps with various logic of CSS SQL html JS php c# py.. prevent the higher efficiency of one logic. All logics can't merge because:

- Unawareness of one logic's efficiency
- Short-term costs / risks of implementation
- Power structure not maximizing efficiency

If app works sufficiently well in separate layers not one logic, it may be not worth to merge. And because one logic requires the insight, skills as well as motivation, it's unlikely to prevail without enforcement. Eg. why people don't speak one language, as it would be useful? Colonial languages often replaced the indigenous ones, as language of the 'masters' was an advantage. The language reform in China 1956, 1st Character Simplification Scheme, Turkey 1932-1982, Korea 1933, Unified Orthography, Vietnam 1910-1945 or Arabic language reform The Nahda Period, Late 19th/Early 20th Century simplified and unified logic to increase literacy / intelligibility. The differencies / specifics slang, accent.. remain due to different regions, cultures, origins.., but the same Spanish is understood in Sevilla, Caracas, Miami.. as English in NY, Singapore, Delhi.... Mandarine in Kunming, Singapore, Taipei.. Russian in Moscow, Almaty, Odesa..

Turing's machine a-machine, 1936 is a metaphor of computer/CPU. Wikipedia says: it "*manipulates symbols on a strip of tape according to a table of rules*" to express any algorithm. The app consists of algorithms of various loops iterations assigning expected outputs for given inputs a-machine, eg. ordering values. Algorithms qualitatively differ eg. by their economies. The less materials chess pieces to construct the same idea in chess composition algorithm, the higher its value. It's the economy and it's generally true: the less iterations energy to express the same idea, the higher its value efficiency. Like the shortest path connecting the same points: eg. route *Moscow-Beijing-Tokyo* is shorter more efficient than *Moscow-Tokyo-Beijing*. Algorithm's quality **Q** relates to number of iterations **i** per its functionality **f**: **Q=f/i** f>=i. The iteration is the **series** of the **same** per unit of time. The same iteration can express intricate as well as trivial thing. Unpointed toes is a minus in gymnastics. The pointed toe is **only 1** unique, unlike **many** unpointed toes' positions: ∑ pointed toes ≪ ∑ unpointed toes. The same flip is less likely uniquer with pointed than random toes =more possible iterations. The lousy style lowers the performace's quality **Q** ↓ as it's divided by more iterations **i** ↑. The high style as pointed toes, clean lines.. reduces the iterations **i** ↓ to increase the quality **Q** ↑. In *Shannon* Entropy 1948, the reduction of the iteration in performance / functionality minimizes the entropy: $-\sum p_i * \log_2 p_i$, $i \in (1,N)$, where N is a set of random positions in flip / performance. The 'pointed toe' is only 1 and its probability is so 1. It gives: $-1*\log_2 1 = 0$ entropy. Any deviation from 'pointed toe' creates N > 1 states of probabilities < 1, with entropy > 0. The random state N_i has probability 1/N for N states: $-N*1/N*\log_2 N^{-1} = \log_2 N$. So the more random states **N** ↑, the higher entropy ⇒ **i** ↑ ⇒ **Q** ↓. Many iterations decrease the quality, only if not utilized in more intricate peformace / functionality. The more iterations per unit of time comparable to CPU's speed, the higher potential usable in high intricacy **Q** ↑ or randomness **Q** ↓.

Pointed Toes, N=1
$\log_2 N = 0$ entropy



Reduction of iterations is called economy in chess composition. As a beginner, I sometimes added unnecessary 'trap' pieces to mislead the solver. I maximized the hardness to solve - a classic criterion in the chess composition masterted by eg. Sam Loyd or puzzles. My 1st published composition intentionally hid the solution by:

- no threat: the mates after necessary move of the black zugzwang
- the solution gives a free flight
- the same mate with distinct tying of the same black Rook by white Queen & Rook
- extra 2 'trap' pawns on a7, b7 to mislead solvers to waste their solving time

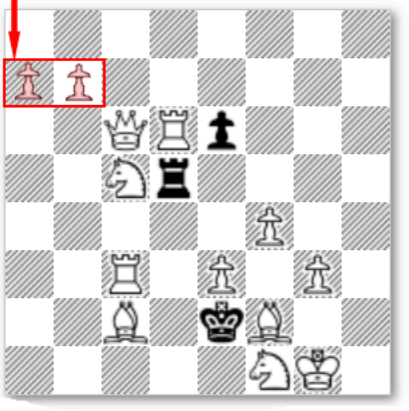
All is unexpected: the zugzwang is rarer than non-zugzwang, free flight is rarer too, the distinct tie in the repeated mate is very rare, and the 'trap' units can slightly confuse solvers. I showed it together with 30 other compositions in March 1994, at Bratislava's circle of chess composition. About 12 present composers couldn't find the solution. One used PC to uncover it. Bedrich Formánek president of FIDE for Chess Composition 1994-2002 removed 2 unnecessary pawns to publish the problem next week in Slovak daily Pravda. I was glad, but thought it was better with the 'trap' pawns that could mislead solvers. Enhancing the options **N**↑, makes the solution less probable harder: 1/N. The extra pieces enhance (+x) the options: N+x, to decrease solution's probability: 1/(N+x) < 1/N. But the 'added value' of the extra options unrelated to the main mechanism, is negligible or detrimental. In quality measure: **Q**=**f/i**, the extra pieces increase both iterations **i** ↑ and functionality **f** ↑ to mislead solvers. If both equally grow: **Δf = Δi** Δ=difference, quality **Q** declines.

In general, **Q=(f+y)/(i+x)**, y=increase of functionality, x=increase of iterations. For equally growing **f** and **i**: Δf=Δi=x ⇒ Q↓, because: Δf=Δi=x, ΔQ=ΔQ/1=(Q+x)/(1+x)=((N-1)/(x+1))+1. For lim x→∞, (N-1)/(x+1)=0, and so $Q_{\lim x\to\infty} = 1$. So for the quality not to decrease, the extra functionality **Δf** must increase more than the extra iteration **Δi**: **Δf > Δi**. 2 extra pawns don't add any special value to componsate the increase of iterations. As a gymnast during a sault could wave or do something special it wouldn't add sufficient value to make a difference. Instead if the energy iteration of 'waving' would be utilized in double sault or series of saults - it would add the higher quality. So the app, design, composition or performance is better **Q**↑ without extra iterations **i**↑, unless they sufficiently enrich the functionality **f**↑↑: **Δf > Δi**.

Brada, Miroslav **author's 1st composition**

Pravda (Bratislava), 25 Mar 1994 (2722)

unnecessary figures removed by B. Formánek



Solution:

1...♔f3 2.♗d1#

1.♗g6!

1...♔f3 2.♗h5#

1...♔d1 2.♗h5#

1...♖d1 (Rd~) 2.♗h5#

1...♖×c5 2.♖d2#

1...e5 2.♗h5#

#2 11+3

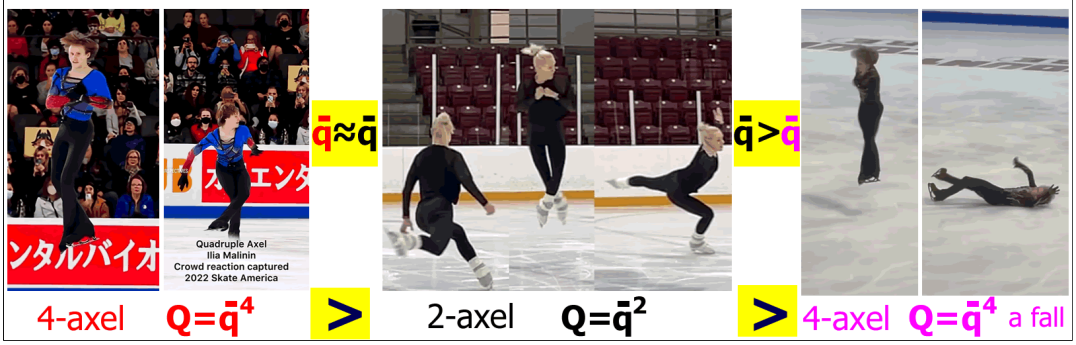
Auto: 2 × Phases, 1 × Twins

Keywords: Flight giving key, Changed mates

N-sault vs Nx saults: $q^N > N*q$

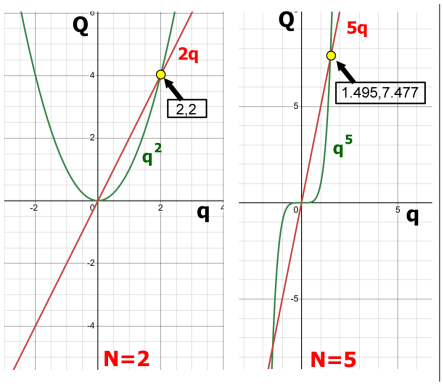
It's easier to sault 2x saults or 1x double-sault? Double-sault includes 1-sault, while some can be able to sault only 1-sault and not double sault. So, N-sault is harder rarer.. than N*1-saults. If the sault's quality is **q**, and the overall quality of **N** saults is **Q(N)** then for **Q(N-sault)** = $\Pi^N q = q^N$, and for **Q(Nx saults)** = $\Sigma^N q = N*q$

Q of N-sault, $\bar{q}$ is average sault's quality: $\bar{q}=(q_1+ q_2+ .. + q_N)/N$



The flawless N-sault no falls, pointed toes.. is always superior to Nx 1-saults: $q^N > N*q$. But any decrease of sault's average quality $\bar{q}\downarrow$ changes the calculus. For **Q(N-sault)** = **Q(Nx saults)** ⇒ $q = e^{(\ln N)/N-1}$ ⇒ $\bar{q}\downarrow$ as **N** ↑. For N=1, the $\bar{q}$ and Q is identical. For {N=2, $\bar{q}$=2},{N=3, $\bar{q}$=1.73},..,{N=10, $\bar{q}$=1.292}..

Overall Quality **Q** grows ●exponentionaly for N-sault: q^N ●linearly for Nx 1-saults: **N*q**



$Q=N*q=q^N \Rightarrow_{/\ln} \ln N + \ln q = N * \ln q$

$\Rightarrow \ln q = (\ln N) / N-1 \Rightarrow q = e^{(\ln N) / N-1}$

N	1	2	3	4	5	...	10	..
q	q	2	1.73	1.59	1.495		1.292	
Q	q	4	5.2	6.35	7.477		12.92	

$\lim_{N\to\infty} e^{(\ln N) / N-1} = 1$

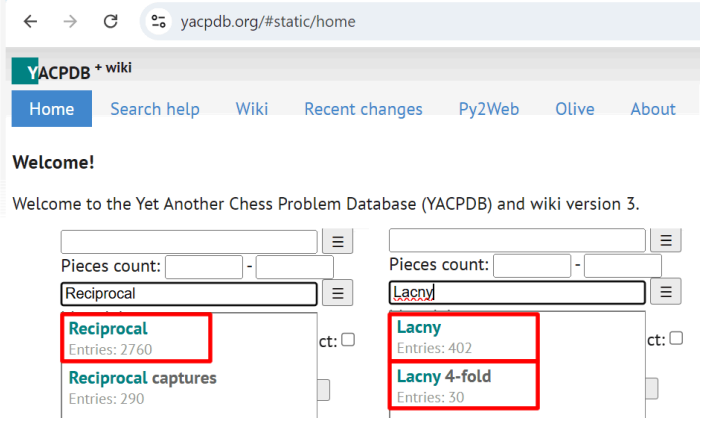
for **Q**(Nx sault) = **Q**(N-sault): **N**↑ ⇨ $\bar{q}\downarrow$

Higher Order of Iterations: $\Pi^N(\Sigma^n \bar{q})$

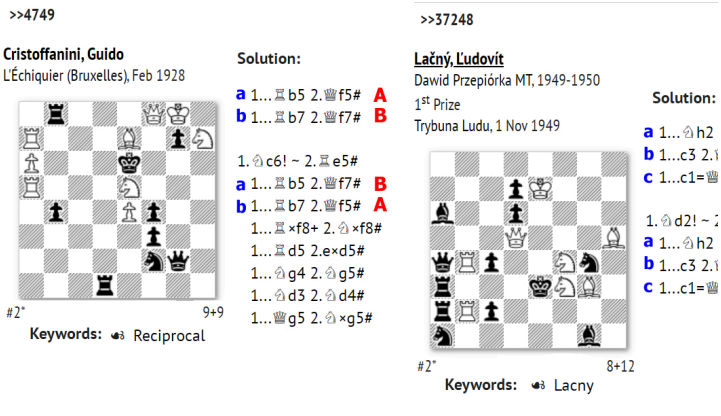
The number of saults in N-sault: $\bar{q}^N$ is Intricacy: **I** ∈ **{1, 2, 3, .., ∞}**, which is higher order of iterations. For Nx saults **I=1**: $N*\bar{q}^1$. So N-sault and N*saults have the same formula: **Q(N)** = $n*\bar{q}^I$, where **N** = **n** + **I**. Or more generally **Q(N)** = $\Sigma^n(\Pi^I \bar{q})$. In a performance of a gymnast, acrobat or figure skater.. the **Q**uality is distributed in various and variously intricated saults/spins. Eg. 4 saults can be distributed in ● 1 sault + triple saults ● 2x double

saults • 1x 4-sault.. Figure skating recognizes 6 types of jumps ordered by difficulty: *Salchow* q_1, *Toe Loop* q_2, *Loop* q_3, *Flip* q_4, *Lutz* q_5, *Axel* q_6, when $q_1 < q_2 < .. < q_6$.. In 1999, Evgeni Plushenko landed 1st a series of: *Toe-Loop* ^{4}+*Toe-Loop* ^{3}+*Loop* ^{2} = **Q(9)** = $\mathbf{q_2^4+q_2^3+q_3^2}$, and in 2002: *Toe-Loop* ^{4}+*Toe-Loop* ^{3}+*Loop* ^{3} = **Q(10)** = $\mathbf{q_2^4+q_2^3+q_3^3}$.
The real Quality must be divided by iterations (N) to get the average $\bar{\mathbf{Q}}$uality per iteration: $\bar{\mathbf{Q}}$**(N)** = **Q(N)/N**. Otherwise many simple jumps, eg. 20x $\mathbf{q_1}$, would exceed **Q**uality of eg. very intricate Quadruple *Axel* = $\mathbf{q_6^4}$ jumped only by Ilia Malinin in 2022. Any series/sequences are comparable by the average $\bar{\mathbf{Q}}$uality eg: • $\bar{\mathbf{Q}}$**(4)**=$\mathbf{q_6^4/4}$ • $\bar{\mathbf{Q}}$**(8)**=$\mathbf{(q_6^3+q_5^3+q_1^2)/8}$ • $\bar{\mathbf{Q}}$**(9)**=$\mathbf{(q_2^3+q_2^3+q_3^3)/9}$ .. etc. The jump's frequency or its 1st date jumped could be used to calibrate its quality **q** relative to other jumps. Or number of falls per jump: the more falls, the harder the jump. I didn't find online stats of the jumps' frequencies in competitons per year, but there is stats of the 1st jumps the later, the harder eg. *Axel* ^{4} 2022, *Flip* ^{4} 2016, *Lutz* ^{3} 2011, *Flip* ^{3} 1981, *Axel* ^{3} 1978, *Lutz* ^{3} 1962.., implying the hardest is *Axel* then *Flip*, *Lutz*. In Olympics 2002, *Flip* had more falls than *Lutz*, although the *Lutz* is considered harder. In our analysis, the figure skating is a useful metaphor and the exact calibration isn't needed.
The higher intricacy, the higher probability of the fall / imperfection: **I**↑ ⇒ $\bar{\mathbf{q}}$↓. A perfection is between 0 or 100%, so the redefined quality is: **q** = **a*p**, **p** ∈ **(0, 1)**, p=perfection. In N-sault both its value **a** and its perfection **p** increase exponentionally: $\mathbf{(a*p)^N}$, while in Nx saults linearly: **N*a*p**. The imperfection **(1-p)** in 1x sault doesn't influence another sault, while any imperfection of the 1st sault of the N-sault propagates to the 2nd until the Nth sault. The level of perfection **p** remains the same for all N saults, while in N-sault it's: p^N, where **p <= 1**. So the N-sault's perfection is: $\mathbf{p^{N-1}}$ times smaller than in N saults.
The mental acrobatics is physically unlimited with far more options. In chess composition the 'sault' is as a change of mate, with enormous other options: change of defences, keys, motifs or paradoxes, hard to imagine in the figure skating. Plus many other stipulations as Mate in 2, 3, .. N, or selfmate, helpmate, studies or numerous fairy definitions. In Mate in 2 moves #2, 2 moves iterations of white lead to the mate to any 1 move iteration of black defence. These 2+1=3 moves iterations can produce a trivial or very intricate composition.
Baghdad's caliph, Mutasim Billah created the oldest known chess problem at c. 840. In Arabic empire, the chess composition was manṣūba مَنصوب. Difficulty to solve had been its main qualitative criterion the harder, the better, until Alberto Mari l'Echiquier Belge in 1928 and Guido Cristoffanini L'Italia Scacchistica in 1927 added the "neo-strategy" criterion in the reciprocal AB-BA change of mates: the mates **A** and **B** to defences **a** and **b** in **phase 1**, interchange in **phase 2**: so the **same** mates **A** and **B** go to defences **b** and **a**. It creates a new unprecedented quality that's a lot harder to construct than just to hide the solution. Ľudovít Lačný made the 1st ABC-BAC cycle in 1945, and 4-fold ABCD-DABC cycle in 1955. The "neo-strategy" employs multiple qualities: $\mathbf{q_1+q_2+..+q_n}$ in phase 1, multiplied by N phases: $\mathbf{Q=(q_1+q_2+..+q_n..)^N = \Pi^N(\Sigma^n q)}$. Or $\mathbf{Q=\Pi^N n*\bar{q}}$. The intricater scheme **I**↑ and more qualities **n**↑, the less likely and later occurs. The chess problem database yacpdb.org listed June 2025 for all stipulations and definitions: 2,760 AB-BA, 402 ABC-BCA, and 30 ABCD-BCDA schemes..

The intricater, the rarer, June 2025



The reciprocal precedes the cyclic change



The formula **Q** = $\mathbf{\Pi^I(n*\bar{q})=(n*\bar{q})^I}$ defines the qualities of all compositions algorithms. Eg: reciprocal AB-BA: $\mathbf{(2\bar{q})(2\bar{q})=4\bar{q}^2}$, Lacny ABC-BCA: $\mathbf{(3\bar{q})(3\bar{q})=9\bar{q}^2}$, where $\bar{\mathbf{q}}$ is the average mate's quality. Mate $\bar{\mathbf{q}}$, key **k**, threat **t**, motif **m** or function **f** can be variously changed / combined in the phases. Any addition within the iterations, without disruptions, increases the overall quality. In my article Chess Composition as an Art, I wrote the Italians composers elaborated the *neo-strategy*: '**a game in a game**' recalling Mannerism High Renaissance, 16C eg. *Arcimboldo*'s collection of objects =phase 1 creating other object =phase 2 as *The Librarian* 1566 from the books. The below table lists the main cyclic themes in the most popular genre in the chess composition: Mate in 2 / #2.

The cyclic changes in orthodox **Mate in 2** until 1999 +2000-2009 | The higher **Quality**, the lower **Number**: Q↑ N↓

Name	Scheme	Year	Author	Q=(n*q̄)^I	Number *		Content
Reciprocal	AB-BA	1927 1928	G. Cristoffanini A. Mari	$4*\bar{q}_2^2$	★1,801	1,549 +252	2 mates **2 phases**
Lacny	ABC-BCA	1949 1950	Ľ. Lačný N. Macleod	$9*\bar{q}_3^2$	224	197 +27	3 mates **2 phases**
4x Lacny	ABCD-BCDA	1955	Ľ. Lačný	$16*\bar{q}_4^2$	17	13 +4	4 mates **2 phases**
Rice cyclic Zagoruiko	AB-BC-CA	1961	J. Rice	$8*\bar{q}_2^3$	35	28 +7	2 mates **3 phases**
Shedey threat Lacny	ABC-BCA	1964	S. Shedey	$9*\bar{t}_2^2$	132	83 +49	threat + 2 mates **2 phases**
Ukrainian** cyclic le Grand	AB-BC-CA	1967 1968	S. Shedey V. Melnichenko	$8*\bar{t}_1^3$	147	110 +37	threat + mate **3 phases**
4x Ukrainian	AB-BC-CD-DA	1978	V. Erokhin	$16*\bar{t}_1^4$	10	9 +1	threat + mate **4 phases**
Ceriani	AB-BC-CA	1981	Ľ. Lačný	$8*\bar{k}_1^3$	1	1 +0	key + mate **3 phases**
Kiss key Lacny	ABC-BCA	1984	I. Kiss	$9*\bar{k}_2^2$	59	48 +11	key + 2 mates **2 phases**
*Reeves	AB-BC-CD	.. / 1985	.. / B. Djurašević	$8*\bar{h}_0^3$	0	0 / 10 twins +0	key + threat **3 phases**
Djurašević	ABC-BCA	1988	J. Rotenberg +J-M. Loustau +M. Caillaud	$9*\bar{h}_1^2$	8	3 +5	key + threat + mate **2 phases**
*4x Djurašević	ABCD-BCDA	.. / 1991	.. / P. Gvozdják	$16*\bar{h}_2^2$	0	0 / 6 twins +0	key + threat + 2 mates **2 phases**
4x Shedey	ABCD-BCDA	1993	Š. Sovík	$16*\bar{t}_3^2$	8	7 +1	threat + 3 mates **2 phases**
☆4x Kiss	ABCD-BCDA	1994	P. Gvozdják	$16*\bar{k}_3^2$	1	1 +0	key + 3 mates **2 phases**
★5x Ukrainian	AB-BC-CD-DE-EA	1998	P. Gvozdják	$32*\bar{t}_1^5$	1	1 +0	threat + mate **5 phases**

source: *Cyclone* (1st edition) and *Cyclone 2* - all known cyclic changes until 1999 and 2000-2009, collected by Peter Gvozdják

★ Stats of "Reciprocal changes" in chess problem database *yacpdb.org*, less precise than *Cyclone*

* **Number** excludes promoted forces, twins, duals, fairy definitions, errors, multiple refutations and problems made after other problems

**Sergey Shedey composed the 1st Ukrainian cycle already in 1967, but published in 1970.

* Reeves and 4x Djurašević don't exist without twins

☆ 4x Kiss has 3 threats repeating in mates ie. duals.

★ 5x Ukrainian with double refutations of tries.

} ⇒ gymnastics' terminology: Both records landed (=no fall) with "unpointed toes"

There are less **keys** (1st moves) than **threats** (2nd moves) that are less than **mates** (2nd move after black defences)

⇒ **Average quality:** $1 <$ mate: $\bar{q}$ $<$ threat+mate: $\bar{t}_1$ $<$ key+mate: $\bar{k}_1$ $<$ key+threat: $\bar{h}_0$

For 1 **key** there are **a** **threats**
For 1 **threat** there are **b** **mates** (↓ black defences)

⇒ $\bar{k}_0 \approx a*\bar{t}_0$, $a>1$
$\bar{t}_0 \approx b*\bar{q}$, $b>1$

x in $\bar{k}_x$, $\bar{t}_x$ is the level of 'dilution' with a mate: $x*\bar{q}$
$x \in \{1, 2, .., \infty\}$ ⇒ $\bar{k}_\infty = \bar{t}_\infty = \bar{q}$

It's the approximation due to heterogeneity and conditionality in chessboard based on figures, functions, rules, defences..

Key **k** is rarer than threat **t** that's rarer than mate **q**, but the composition needs a threat (except zugzwang that's rare). So the threat cycle (Shedey, Ukrainian) can be easier to construct even though the threat paradox (in threat cycles) is rarer than change of mate.

⇒ No threat schemes: $Q=(t+n*\bar{q})^I$, $t \approx x*\bar{q}$ ⇒ $Q=((x+n)*\bar{q})^I$

It explains why there are **147** *Ukrainian* cycles: $8*\bar{t}_1^3$ and only **35** *Rice* cycles: $8*\bar{q}_2^3$ ⇐ $\bar{t}_1 < \bar{q}_2$

Due to added threat **x**:
Rice's cycle quality: $((x+2)*\bar{q})^3_2$
Lacny's cycle quality: $((x+3)*\bar{q})^2$
for x=(1/2) ⇒ $Q = \frac{15.625*\bar{q}^3}{12.25*\bar{q}^2}$ instead of $\frac{8*\bar{q}^3}{9*\bar{q}^2}$

Lacny has only 2 phases, so the quality of *paradox* in the *Shedey* cycle can overcome the threat in *Lacny* (no-threat) cycle, so the diference in numbers isn't big: **224** *Lacny* vs **132** *Shedey*. No surprise, non-threat cycle prototypes: Cristoffanini (1927), Lacny (1949,1955), Rice (1961), are in a set-play form (=no threat in phase 1) easing their construction. The compositions using the checks as the kyes don't need threats and so are easier to do: they are 9≈31% of Rice (until 1999), 4≈2% of 3x Lacny, and 3≈23% of 4x Lacny. The varios other analyzable criteria of the quality exist: free flight, non-check keys, non King defences..

The Librarian, 1566

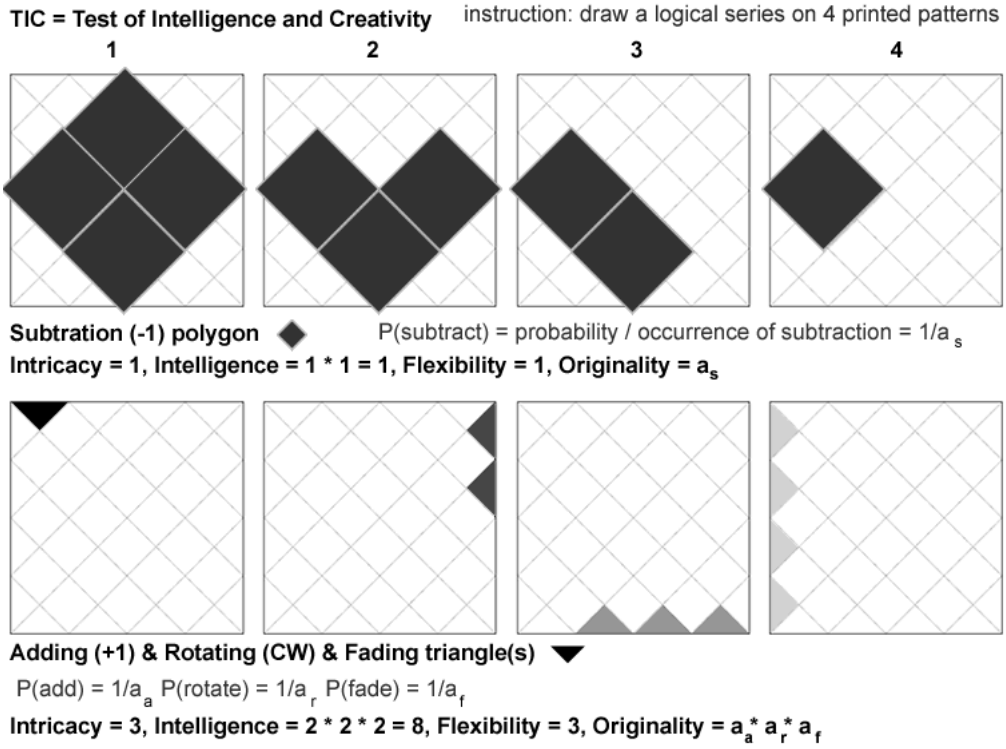
Data of #2 cycles is representative, although not 100% complete. A few harder schemes: *Rice* and *4x Lacny* preceded the easier: *Shedey*, *Ukrainian*, *Kiss*. It's because *Rice* and *4x Lacny* are logical continuation or extension of the 1st cycle: *3x Lacny*, and so the composers had been primarely focusing on them. But later between 2000-2009, there were significantly more *Shedey* (49), *Ukrainian* (37) than *3x Lacny* (27). Also the threat cycles include the 'threat paradox' shown by A. Dombrovskis 1958, so the threat cycles couldn't appear before 1958.
The higher quality, the lower occurence holds in gneral eg in: Selfmates, Reflex mates.. or longer #3, #4.. Eg. only 1 Lacny and 1 Kiss cycle exist in #4, which isn't only about difficulty but less composers focused on it. So data of #2 is the most reliable to analyze. The number of 3x Lacny since 1950 in decades: 1 1949, 27 50s, 49 60s, 49 70s, 31 80s, 40 90s, 27 2000s. Peaking in 60-70s. Set of new schemes is finite: falling to 0 by time. By 2009 3x Lacny in orthodox #2 could be 90+% exploited. The cycles occuring later Shedey, Ukrainian.. peaks later.

And there is a bigger picture. Chess composers are much smaller group than chess players as all chess composers can play chess, while not all chess players can do compositions. Since the end of 90s, the youth has become less attracted by the chess composition probably by chess too, due to the rise of PCs, internet / social media absorbing attention, and a higher commercialization. I started composing the chess problems shortly after the fall of socialism in Czechoslovakia in the begnning of 90s. Most newspapers had a regular chess problem column with composing competitions. After the regime change, the chess columns were gradually disappearing, while the tabloids with topless women arose.. As a Phd student of economics I elaborated the idea of "redirection of talent" to explain the rise of chess composition in Slovakia or other then socialistic countries, overcoming the West. In this view, talented people were "captured" in "alternative activities" as chess composition as they couldn't pursue the productive activities. I knew it was to fit the "market economy" ideology - omnipresent after 1989. But, the decline of interest in the chess composition in the late 90s has happened at the West or India too. And the Czech Republic and Slovakia instead of developing the new technologies (utilising the "supressed talent" before 1989), has lost know-how eg. in heavy and processing industries or agriculture. And the know-how has declined since 1990s also at the West mostly due to outsourcing, technologically overtaken by China / Asia. That's why the rise of the chess composition in the former socialistic countries is rather the spillover effect of then Zeitgeist emphasizing constructivity - despite some restrictions (and all systems have their own restrictions).

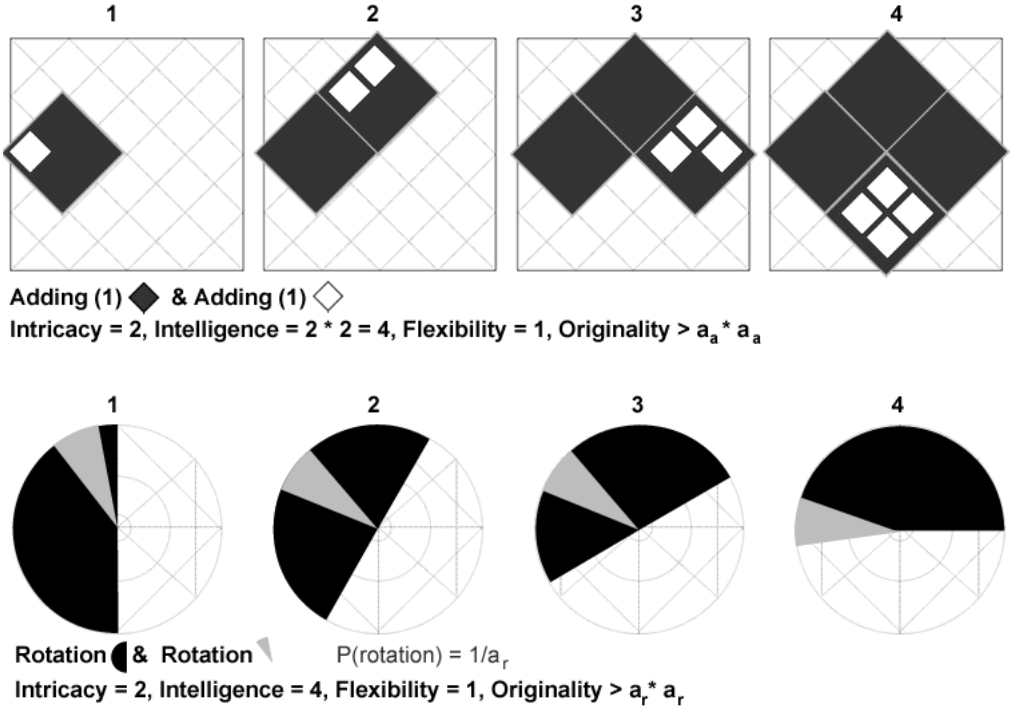
Options, Logical Series and Intricacy

Deep Blue computer beated then the best chess player *G. Kasparov* in 1996. But until now 2026, no computer / program or AI can compose the cyclic shifts. Algorithm / AI can test all positions to identify the cyclic schemes. It seems unreal not only due to too many options c. $(4.822\pm0.028)*10^{44}$ Tromp + Österlund, 2025, but also difficulty to write identification algorithm. *Shannon* number 1950 estimates c. 10^{120} chess positions exceeding the atoms in the observable universe c. 10^{80}. Quick estimate is: 32 pieces on 64 fields: $32!*64! \approx 10^{124.52}$ minus the illegal positions as pawns on the 1st fields, king next to the enemy king... The cyclic #2 can further exclude eg. less than 6 or more than 30 pieces.. But the chess problems can use a huge set of fairy pieces Alfil, Camel, Vao.. or rules Circe, Patrol, MAFF.. with infinite options. T.R. Dawson 1912 defined Grasshopper moves along ranks / files / diagonals by hopping over another piece. P. Monréal 1967 Circe captured pieces reborn on their starting squares, A.J. Karwatkar 1979 Madrasi 2 pieces of the same type & opposite color attacking each other, can't move, capture, give check.. In 1995, I found a new class of the rules redefining the mate: MAFF mate with a free field, OWU one white unit in the black king's field or mating conditions: AMU mating unit must be attacked before moving, UIA the same color unit must be in the activity of the mating unit.. I summarized them in article New Ideas in Chess Composition in the American journal StrateGems 2001, with examples as total 3x3 *Shedey* impossible in orthodox #2. Mate with "1" free field MAFF is one option of *n-MAFFs*: mate with "n" free field(s) $n\in\{1,..,8\}$, as *n-OWUs*, *n-UIAs*.. It reveals the mate or chess itself as an convention definable otherwise.
In 2017 in Kyoto, I met Tadashi Wakashima, a chief editor of the Japanese magazine Problem Paradise, where I had published a few problems, and noticed the section of Japanese chess Shogi 将棋. To my surprise, Tadashi told me the Shogi compositions are impressive as in the chess. It has 9x9 board with 40 pieces incl a few specific ones Generals, Lances and captured pieces can be dropped back onto the board similar to *Circe*. Quick estimate is: $40!*81! \approx 10^{168.67}$ positions, legal c. 10^{69} Miyako + Kiya + Ono, 2002 - more than chess and the returning pieces further increases Shogi's options. Indian 7th cent. Chaturanga चतुरङ्ग developed to Shatranj شطرنج in Persia, inspired Shogi, Chinese / Korean / Burmese or Thai / Cambodian chess.. Some claim the Chinese chess Xiangqi 象棋 had been the 1st. The oldest known game, Chinese *Go* 围棋 / wéiqí c. 2,000 BC considered unprogrammable due to vast c. $2.08*10^{170}\approx10^{170.32}$ options on 19x19 board, Tromp + Farneback, 2016 , until **AlphaGo** beated then the best *Go* player *L. Sedol* in 2016. Quick estimate is (2 pieces + 1 void): $3^{19*19}=3^{361} \approx 10^{172.24}$ - more than in chess but only due to the bigger board. On 8x8 chessboard *Go* has max $3^{64}\approx10^{30.54}$ options - far less than chess, so per board's unit chess has more options due to more pieces definitions. The mere 2 types of *Go*'s pieces white / black can't create schemes as #2, #3.. or selfmate, helpmate.. It enables only compositions known as *Tsumego* 詰碁 comparable to chess endgame studies.
The N*1-salts is easier to perform than 1*N-salt. Similarly, logical series of IQ test can have N*1-logics or 1*N-

logics. In my Master thesis in psychology 1998, I made a test *TIC* Test of Intelligence and Creativity to assess series drawn on the patterns. It has 4 different patterns and 16 series of these 4 patterns with 4 iterations. It's an experimental method to complexly study the IQ and its interrelations. Instead of selecting the right solution classic IQ test, the series is created to capture the IQ, creativity originality, flexibility.. and projected personality traits. The valid series contains the repeated sign(s) =iteration eg. rotation, analogy, sum.. Testing 600+ people, I identified 24 distinct logics from the frequent as adding, alternating to the rare as fading..

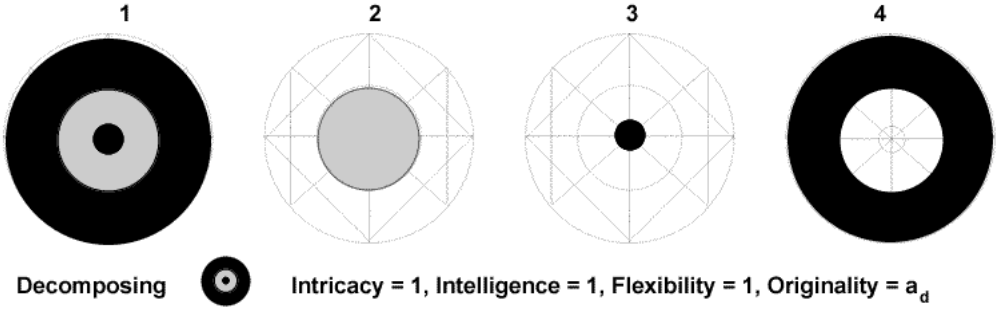


Both N*1-logic and 1*N-logics have the same number of iterations, but the score is exponentionally higher for the combined logics. Flexibility: the distinct number of logics, increases the options to combine logics, so intelligence correlates with flexibility but not 1:1, because the same logic can be combined repeatedly into one intricater series. Eg. below is example of Adding white square into black Adding square or Rotation in Rotation.



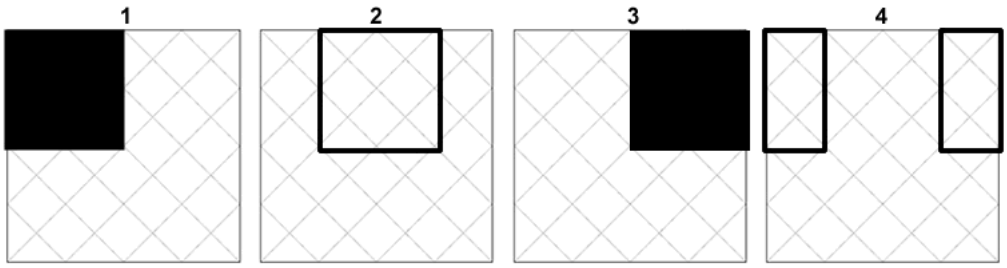
The intricater series, the higher originality, when the inverted probablities of the logics multiply. So the intelligence correlates with originality with psychological specifics. Eg. the intricater series combining the same series as Add & Add & Add.. should be less original than eg. Rotate & Move & Alternate.., as the 'adding' is the most common least original logic. In fact, the intricater series with the same logic is rarer than the combined distinct logics. If somebody starts drawing 'the adding', it's less likely to get idea to use another adding to 'the adding', than eg. to rotate or move 'the adding'. As in a math joke: how to decrease the probability of the bomb in the airplane? To bring the

bomb! Because the probability of two independent bombs is multiple smaller than 1 bomb.
The same logic creating N-series absorbing all iterations explain psychosis. Like a recursive endless loop in the code absorbing the whole time / memory to halt the whole system. I clarified this mechanism in part of my Master thesis 1998 as Model of Schizoprehnia. Self-identity is a logical series of iterated 'ME': ..ME-ME-ME-ME.. in time and space. If paranoia aborbs the all iterations, it consumes self-idenity's iterations too, inevitably ending in psychosis. The content of paranoia is irrelevant God, Martians, computer.. what matters is the repeated logic eg: 1) "they observe me", 2) I know "they observe me", 3) they know "I know "they observe me"", 4) I know.. etc.. The logic continues until the last iteration.
Although very intelligent persons can suffer from schizophrenia, the probability of schizophrenia decreases with intelligence, as the higher intelligence =more iterations per unit of time, the less likely the all iterations would stay in 1 series. I used a N-die metaphor, where number of throws is iterations: the higher intelligence, the more iterations. Schizophrenia is like a roll a die would always give 1 series eg: 5, 5, 5, 5.. or 1, 1, 1, 1.. Bi-polar manic-depressions is like 2 series eg: 5, 5 .. and 1, 1.. or 6, 6 and 2, 2.. N-sided die enables N schizophrenias, and N*(N-1)/2 bi-polars. N are distinct options logics that differ in societies / situations. Possible personalities for N options are: N^{IQ}. Two conclusions are: 1. Probability of schizophrenia is: $1/N^{IQ-1}$, of bi-polar is $(N-1)/2N^{IQ-1}$ Both decrease exponentionally with IQ, and linearly with options. 2. Bi-polar is (N-1)/2 times more often than schizophrenia. Societies with more options eg. richer / more advanced, have higher ratio of bi-polar with respect to schizophrenia.

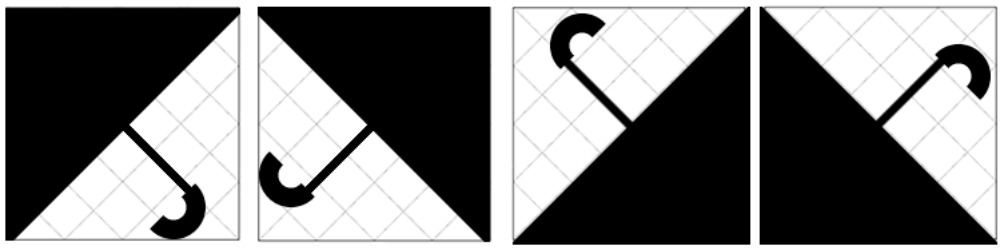


Decomposing **Intricacy = 1, Intelligence = 1, Flexibility = 1, Originality = a_d**

The profession or identity also affects the likelihood of logic. Decomposing is rarer logic in general, but it has been used quite often by people in technical fields, where the problems to be resolved are decomposed to smaller / simpler units. A typical example is integration by parts Brook Taylor, 1715.
The TIC test can capture various aspects of the projected logic eg. meanings or effects. "Meaning" is irrelevant to logic but adds something extra as a rotating umbrella. "Effect" is eg. re-appearing or bouncing of the drawn unit.



Alternating (full, empty) & Moving (right) + effect 'eternal': moving square reappears in the beginning
P(alternating) = $1/a_l$, P(moving) = $1/a_m$, P(effect eternal) = $1/a_{ee}$
Intricacy = 2 + 0.5 (effect), Intelligence = 2.5^2= 6.25, Flexibility = 2.5, Originality = $a_l * a_m * a_{ee}$

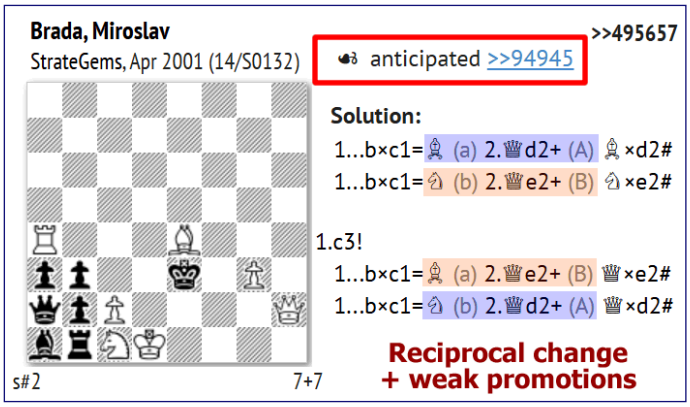


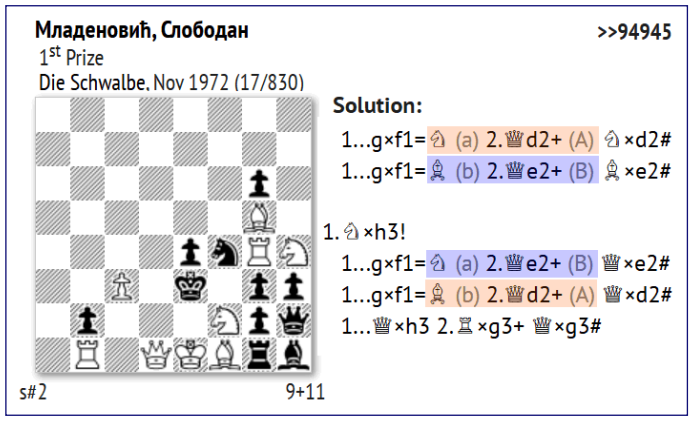
Repeating & Analogy + effect 'meaning': umbrella
P(repeating) = $1/a_p$, P(analogy) = $1/a_n$, P(effect meaning) = $1/a_{em}$
Intricacy = 2 + 0.5 (effect), Intelligence = 2.5^2= 6.25, Flexibility = 2.5, Originality = $a_p * a_n * a_{em}$

Paradox of uniqueness

Calculus in the 17th century was uniquely intricate series. How is possible that *Newton* and *Leibniz* invented the calculus independently near the same time? The uniqueness grows with the intricacy, and chances of multiple discoveries raise too, as the intricater idea, the less ways how to do it. It's observable in chess composition with highly intricate schemes. I composed c. 200 chess problems between 1993-2000, often reciprocal or cyclic schemes. I discovered the same idea without knowing of renowned composers 3-times: *V. Rudenko*'s #2 Lacny 1958, *S.*

Mladenović's #2 reciprocal self-mate 1972, and *J. Valuška*'s #2 Shedey 1992. Below is the recurred self-mate. The editor - the prominent composer *M. Vukcevich* didn't recognize the same mechanims from 1972 to write: "This is Brada's first selfmate and it is probably unique". The S#2 with reciprocal shift + weak promotions was my 1st self-mate and last problem, done in Dec 2019.

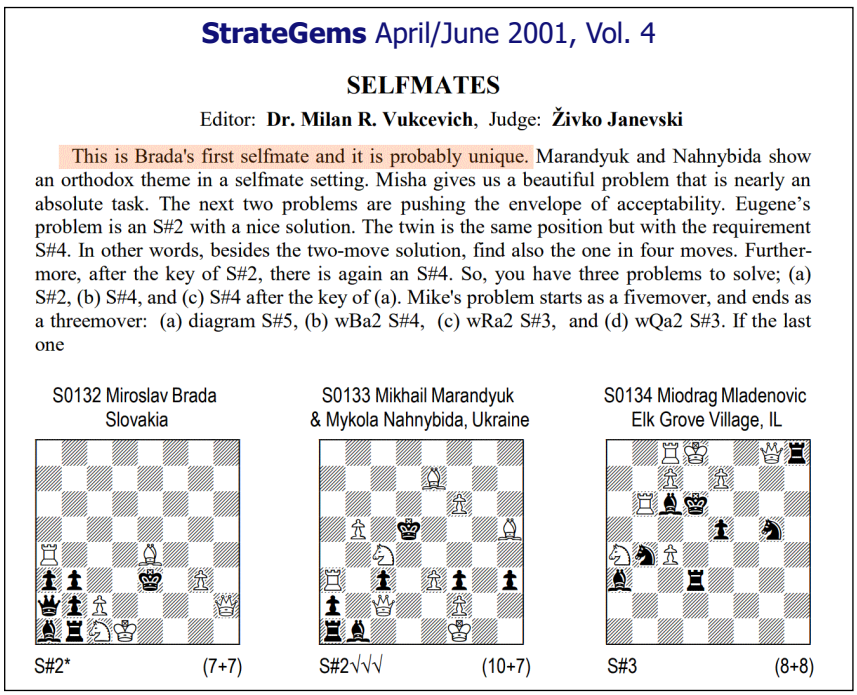




Paradox of uniqueness has surprising insights. Eg. poetry seems unique as it's improbable that the famous poets as *Pushkin* Пушкин, *Goethe*, *Shakespeare*, *Ferdousí* فردوسی or *Li Bai* 詩仙 would write the same poem. "Same" doesn't mean the exact words, but the plot / mechanism.. Since there are more ways to write poem than to define eg. calculus, poetry is unexpectedly less unique than calculus, and so paradoxically less likely to recur.
Or: infinitely intricate "**∞**-series **S**" is unique as it has only 1 way how to do it. If the other person creates "**∞**-series **T**", it's identical to "**S**": **S==T**, as it has only 1 way. So the **∞**-series must recur, but because nobody can create **∞**-series, it never recurs.

Title		=	i↑	i=0	i=∞
Probability of **i**-series	**P(i)**	$\mathbf{1/a^i}$	↓	**1**	**0**
Set of **i**-series	**M(i)**	$\mathbf{M^{1/i}}$	↓	**∞**	**1**
People able to do **i**-series	**N(i)**	$\mathbf{N/x^i}$	↓	**N**	**0**
Recurrence of **i**-series	**R(i)**	**N(i)/M(i)**	-	**0**	**0**

The recurrence rate **R** = $\mathbf{N/(x^i(M^{1/i}))}$ is tricky as **M** and **N** are interlinked. Economist M. Kremer in Population Growth and Technological Change 1993 finds a bigger population in bigger areas boosts the innovations spillover and scale effects to enable more people to thrive. As a result, the bigger areas eg. India / China.. have been more densely populated than smaller isolated areas as Tasmania.. So the number of series inventions **M** grows with population **N** that grows with the number of series **M**: .. ⇒ **M**↑ ⇒ **N**↑ ⇒ **M**↑..


StrateGems April/June 2001, Vol. 4

SELFMATES

Editor: **Dr. Milan R. Vukcevich**, Judge: **Živko Janevski**

This is Brada's first selfmate and it is probably unique. Marandyuk and Nahnybida show an orthodox theme in a selfmate setting. Misha gives us a beautiful problem that is nearly an absolute task. The next two problems are pushing the envelope of acceptability. Eugene's problem is an S#2 with a nice solution. The twin is the same position but with the requirement S#4. In other words, besides the two-move solution, find also the one in four moves. Furthermore, after the key of S#2, there is again an S#4. So, you have three problems to solve; (a) S#2, (b) S#4, and (c) S#4 after the key of (a). Mike's problem starts as a fivemover, and ends as a threemover: (a) diagram S#5, (b) wBa2 S#4, (c) wRa2 S#3, and (d) wQa2 S#3. If the last one

S0132 Miroslav Brada
Slovakia
S#2* (7+7)

S0133 Mikhail Marandyuk
& Mykola Nahnybida, Ukraine
S#2√√√ (10+7)

S0134 Miodrag Mladenovic
Elk Grove Village, IL
S#3 (8+8)

During my 1-year grant research 1999 I tested 600+ persons with various tests including *TIC* to draw logical series. I found 20 (=M) distinct **1**-series, which wouldn't increase even I'd test 100x more people. To determine the recurrence rate for the intricacy is to answer: Does a population able to create series **N(i)** falls more or less than a number of series solutions **M(i)**, as the intricacy grows **i**↑? *TIC* captures the IQ and creativity to allow any intricacy: **i∈(0,∞)**.

The *TIC* series are combinable in $\mathbf{M^i}$ distinct **i**-series, diverging **M**↑↑ as **i**↑. And a population able to create **i**-series decreases as **i**↑ ⇒ **N**↓. So the recurrence in *TIC* decreases exponentionally **R**↓↓ as **i**↑: **Δi** > **ΔR(i)**. But the real puzzles / inventions have a finite set of **i**-series **M**, and big discoveries as a wheel, analytic geometry, calculus, alternating power system, converge to 1 series: **M**→**1**.

For **M**=**1** ⇒ $\mathbf{M(i)=1^{1/i}}$ ⇒ the set of series is always 1, regardless of intricacy **i**, so 1 solution can be trivial as well as sophisticated. A wheel recurred independently in Mesopotamia c. 4,000 BC c. 1-2M people, Eastern Europe

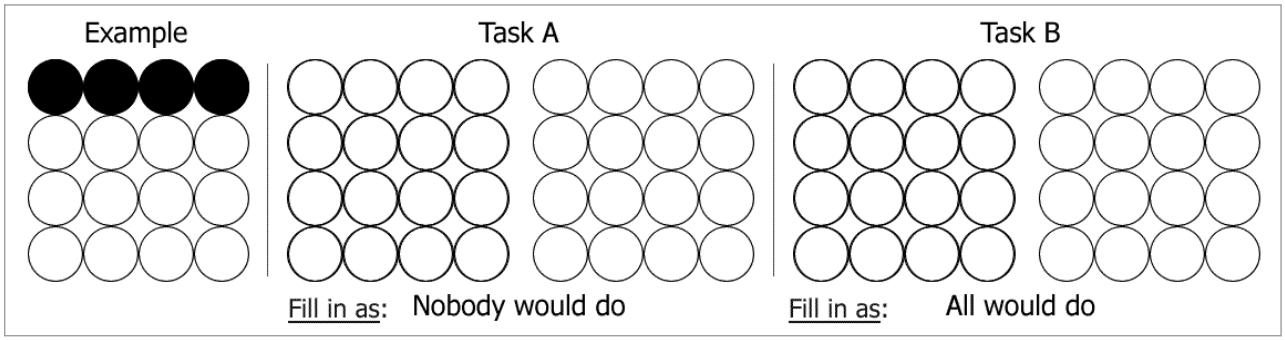
Carpathian Mountains c. 3,900 BC c. 1M people, China 2,800 BC 100K - a few millions, Americas 1,500 BC Mesoamerica c. 0.2-1M. Data suggests c. 1M people is a threshold to wheel's invention. In the 17C, when calculus arose, *Newton*'s England had c. 5.5M, *Leibniz*'s Holy Roman Empire 15-20M. Only a fraction of population counts, depending on opportunities, abilities, education, motivation to invent. Eg. literacy in Mesopotamia was <10%, while in 17 c. Europe ≈30% ≈10% women. The wheel wasn't invented suddenly by 1 person, but gradually spanning a few centuries. Calculus also used the earlier ideas of *Descartes* 17C, *Euclid* 300BC.. So the population involved in inventions cummulates the previous populations, while only its smaller fraction can invent. To simplify, the population **N** at time of invention can be used, and $\mathbf{x}=\mathbf{2}$ in $\mathbf{N(i)}=\mathbf{N/x^i} \Rightarrow \mathbf{N(i)=N/2^i} \Rightarrow \mathbf{i}=\mathbf{ln_2(N/N(i))}$, "x=2" means ≈50% people can create **1**-series invention to dwindle by $\mathbf{2^i}$ as invention's intricacy grows **i**↑: 1/4, 1/8.. Then population per invention **N/N(i)** determines the intricacy: the wheel appears in $1M=10^6 \Rightarrow \mathbf{i}=\mathbf{ln_2\ 10^6} \Rightarrow \mathbf{i}\approx\mathbf{19.93}$. The calculus in 5.5M 17 c. England ⇒ **i≈24.7**. In 2003 *G. Perelman* from St. Peterburg proved in 7 years the *Poincaré* conjecture, a century-old math problem, to be more famous after refusing $1M Millennium Prize. Russian 2003 population was ≈145M ⇒ **i≈27**. These calculations are illustrative to show the invention's intricacy always grows with a population / time, since the current inventions cummulate the earlier ones. But the invention's uniqueness is its change from the initial conditions / time: $\mathbf{(i_t - i_{t-1})/\Sigma^{t-1}\ i}$. *Newton* couldn't invent calculus in Mesopotamia or the 17C. Australia, and maybe he wouldn't invent the wheel either. The wheel could be greater change than modern inventions - but maybe not. But the later the invention, the bigger change must be added to be equally significant as the earlier inventions. Eg. the uniqueness of the change=1 for the 1st invention is: 1/0=∞, 2nd: 1/(1+1)=1/1, 3rd: 1/2 etc.. So the invention's uniqueness falls in time cummulating earlier ideas, unless its change doesn't sufficiently grow. Google AI 28 June 2026 answers: "Is *Perelman*'s proof more unique than invention of calculus?"

> *"No, Perelman's proof isn't mathematically more unique than the invention of calculus, but it's vastly more unique in its modern historical and psychological context. To compare them accurately, we must look at how mathematical ideas are born. Calculus represents a fundamental paradigm shift that created an entire branch of mathematics, while Perelman's proof of the Poincaré Conjecture represents a pinnacle of synthesis, bringing together existing, disparate fields to solve a century-old problem."*

The math inventions set of series have hugely grown **M**↑ since the 17th C., some estimate by 95%. *Perelman* combined 5+ math areas, most of which hadn't existed in the 17th C: Differential Geometry 17th C., Geometric Analysis 1930, Geometric Topology 1935, Alexandrov Geometry 1941, Ricci flow 1941.. The overall set of series of the combined N fields isn't $M_1+M_2+..$, but $M_1*M_2*.. \Rightarrow \mathbf{\Pi^N_{i=1}M_i}$. The new invention multiplies the previous options to expand set of series **M** more than linearly, so 95% growth creates far more new options than it can look. The population has also grown **N**↑ ≈11x in time **t** ≈550M 17C to ≈6.1B 2003, but less than series **M**↑↑: $\mathbf{\Delta_t M > \Delta_t N}$. So the recurrence rate falls on average $\mathbf{R}=\mathbf{\Delta N/\Delta M} < \mathbf{1}$, as population grows **N**↑.

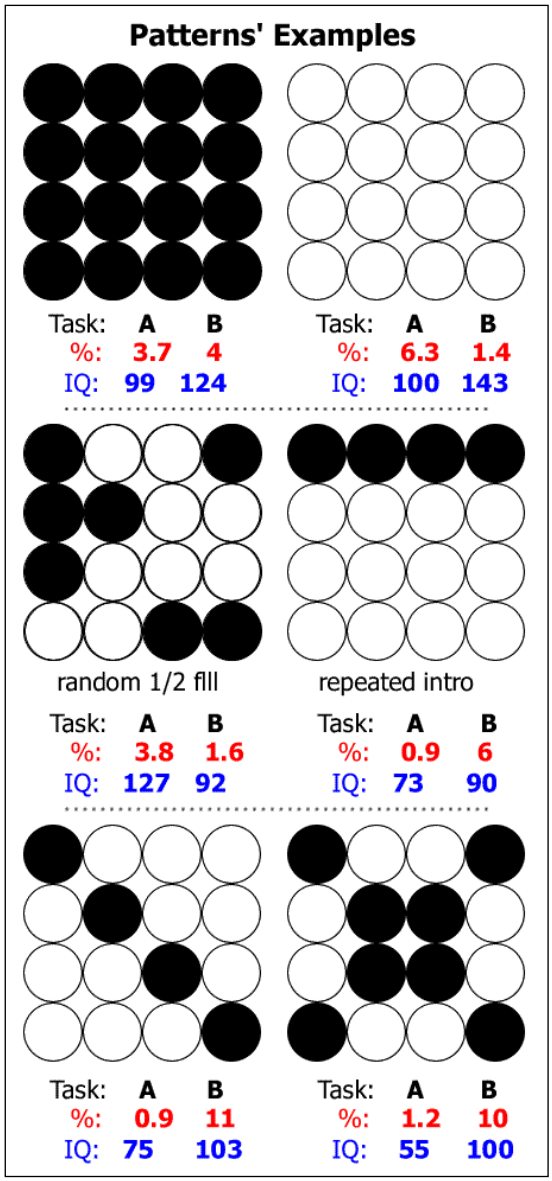
The 3x3 *Rubik*'s Cube 1977 has $43*10^{18}$+ options. Some estimate ≈1% find one of 4 known solutions series ⇒ $\mathbf{M(i)}=\mathbf{4}$. For $\mathbf{x}=\mathbf{2} \Rightarrow \mathbf{i}=\mathbf{ln_2\ 100} \Rightarrow \mathbf{i}\approx\mathbf{6.64}$ and set of series $\mathbf{M}\approx\mathbf{10,002} \Leftarrow \mathbf{M} = \mathbf{4^{6.644}} \Leftarrow \mathbf{M^{1/6.644}}=\mathbf{4} \Leftarrow \mathbf{M^{1/i}}=\mathbf{4} \Leftarrow \mathbf{M(i)}=\mathbf{4}$. Mere 4 *Rubik*'s solutions indicate the 25% recurrent rate for ≈**1%** of population. The *CFOP* solving method series is the most common, so the recurrence can differ for each solution, but still growing with the intricacy: $\mathbf{\Delta R > \Delta i}$, as for M≈10,002, there are ≈9,998 wrong series tried by ≈99% people, versus 4 solutions tried by 1%. It's the 25x higher recurrence: $R_{right} \approx 1/4 = 0.25 > R_{wrong} \approx 99/9,998 = 0.01$.

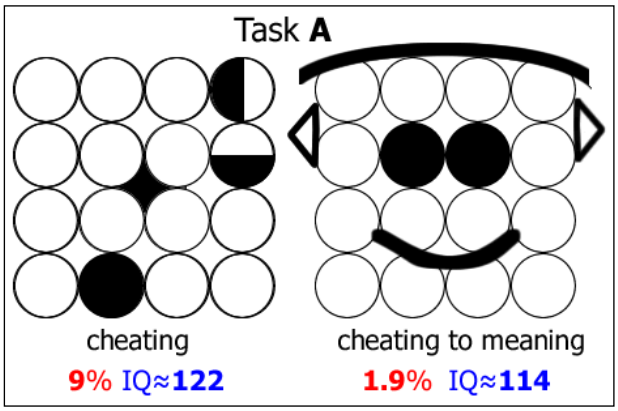
In my 1999 research, I used *TIC* and other own tests eg: "Guess other Guess", asking to fill in 4x4 circles, 2x as "Nobody would do" and 2x as "All would do". The example showed 4 upper filled circles.



16 circles have 2^{16}=65,536 patterns. The best in *Task A* is to randomly fill in half (8) circles to maximize options: C(16,8)=12,870, to minimize matches. It was used by av. IQ ≈127 assessed by *TIC*. The best in *Task A* is 1 option ie. to fill all {16} or none {0}: C(16,0|16)=1, used by IQ ≈124 for {16}, IQ ≈144 for {0}. The worst for *Task A(B)* is the best for *Task B(A)*. The *Task A*'s worst {0,16} patterns had IQ≈100. *Task B*'s worst 8 fill random patterns had IQ≈92.

Testing 600+ people showed '8 full random' patterns in *Task A* were rarest. But a decision to use '8 full random' was more often ≈3.8% than some 4-full symmetric patterns < 1%. The commonest pattern in *Task B* wasn't the {0,16} patterns, but the symmetric (4 full) patterns, having IQ ≈100. So *Task B*'s best choice wasn't the "best" as the majority filled sub-optimal symmetries. The super intellect IQ→∞ could "guess" the majority won't fill the best patterns to mimic the "best" guess, but the sub-optimal guesses diverge. In Task A, some "cheated" (drawing outside), achieving unique patterns, but the incentive to cheat isn't so unique used by 9% of IQ≈122. 1.9% extended "cheating" to create a meaning as face, used by 1.9% of IQ≈114. The repeated intro example with 4 full upper circles isn't optimal choice for none of the Tasks, used so by lower IQs. The *TIC* allows higher variability, ie. *TIC*'s IQ≈55/73.., would score higher by standard IQ tests.

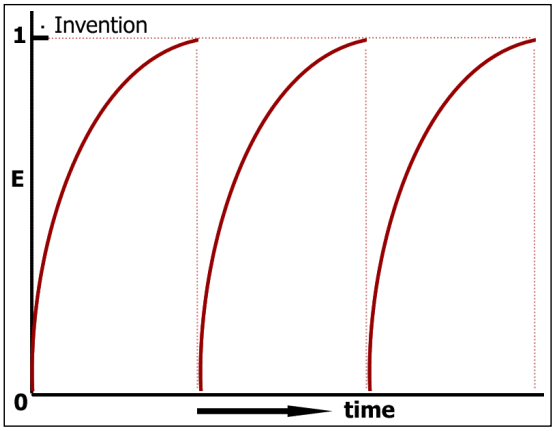




"Guess other Guess" confirms the choice's quality grows by IQ / intricacy. The high IQ can miss the majority's suboptimal guess to doubt efficiency of a collective choice. Many choices are easy, all, regardless of IQ, prefer clean air, water.. or healthy food.. But some specific tasks are unresolvable by the majority's choice. It then depends on the quality of the criteria and motivation to assign the right experts. Eg. oligarchies (often masked as democracy) miss motivation to optimize the public choice. Or some institutions may prevent promoting the ablest, to rid of too able rivals.

The population growth **N**↑ multiplies spillover & scale effect the logical series **M**↑↑ to outgrow and boost the population growth: **ΔN** < **ΔM** ⇒ ..**N**↑ ⇒ **M**↑↑ ⇒ **N**↑ ⇒ **M**↑↑ ⇒.. The IQ / intricacy **i** reduces a number of series $\mathbf{M(i)=M^{1/i}}$ to 1 or few solutions: **Δi** > **ΔM** ⇒ **i**↑ ⇒ **M**↓↓ ⇒ **M**→**1**, while growing intricacy **i**↑ can lead to invention - a new set of series $\mathbf{M_n}$ to multiply the overall set **Δi** < **ΔM** + $\mathbf{M_n}$ ⇒ **i**↑ ⇒ **M**↑↑. So the formula $\mathbf{M(i)=M^{1/i}}$ to be complete must include expectation **E** of the invention $\mathbf{M_n}$ growing with intricacy **i**↑ ⇒ **E**↑: $\mathbf{M(i)=M^{1/i} + E(i)*M_n(i)}$, where **E**∈**(0,1)**, **i**→**0** ⇒ **E**→**0**, **i**→**∞** ⇒ **E**→**1**. It can be: $\mathbf{M(i)=M^{1/i} + (1\text{-}2^{-i})*M_n(i)}$.

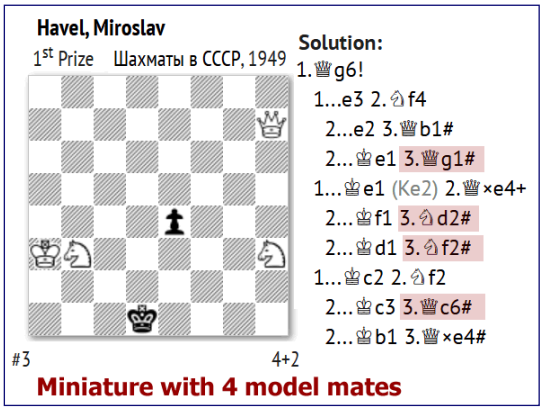
The recurrence **R**=**N(i)/M(i)** dynamically changes to be the most likely after the set of series **M** is exploited **M**→**1**, which is a period exactly before the invention - when it quite likely recurs as: calculus 17C., bakelite 1907: *Leo Baekeland* / *James Swinburne*, light bulb 1879: *Joseph Swan* / *Thomas Edison*, photography 1830: *Louis Daguerre* / *William Talbot*, telephone 1876: *Alexander Bell* / *Elisha Gray*.. In contrast, the periods after the invention are the least likely to bring the new significant invention to match "Paradox of uniqueness": the uniquer inventions, the higher probability of their recurrence. Of course, there are inventions as *Perelman*'s proof not re-discovered by anyone else, but the multiple discoveries are more often than "common-sense" would expect - as the inventions need opportunity and high intricacy excluding the majority.



Interestingly, all of my 3 recurred compositions were of the Slavic composers. Moreover *Ján Valuška* is from my home town Zvolen in Slovakia, although we weren't meeting, and *Valentin Rudenko* was from Dnepropetrovsk, the home city of my grandmother. As a leading designer in Yúzhnoye Южное, the main soviet aerospace and defense manufacturing facility, his team constructed 60+ science satellites. Finally Slobodan Mladenovic' was one of the

most distinguished Yugoslav Serbian composers. The recurred schemes of composers speaking Slavic languages Slovak, Russian, Serbian.. could speculatively indicate that certain ideas / associations are likelier in certain linguistic groups. The cyclic changes are more common in Slavic spoken countries, when Slovakia is per capita the most "cyclic nation" with several cyclic prototypes. Nevertheless, many cyclic problems arose by non-Slavic spoken composers including Indians. 3 recurred compositions isn't a big sample, so they could happen randomly, but an underlying logic can be explored. A declension inflection of nouns / pronouns / adjectives is typical for all Slavic languages except Bulgarian / Macedonian with the changing endings. Eg. in English you say: "a rose", "of a rose", "give a rose", "about a rose", "with a rose", while in Slovak: "ruža", "od ruže", "daj ružu", "o ruži", "s ružou" - that's particularly difficult for non-Slavic speakers to learn. And the inflection of the same word ressembles a changed mate / function in the cyclic or reciprocal changes. One of the most prolific chess composer in history Bulgarian *Petko Petkov* with the most points for his compositions in the FIDE albums - never constructed the cyclic shift of mates. Bulgarian is the only Slavic language without declensions, which could explain a lot less cyclic problems on average done by Bulgarians in comparison with other Slavic nations. Surely, there are more influences forming the spread of certain logic, but the possibility that some types of schemes are likier in certain linguistic / cultural groups can't be excluded, and the evolution of the chess composition enables to study it.

Evaluation of Art $(\Sigma^{N}_{n=1} A_n^2)^i$



In Czech journal Světozor 1869, the painter and chess composer *Antonín König* described the chess composition as an independent form of Art to initiate the Bohemian School Česká škola úlohová, the 1st art movement elevating chess composition beyond the mere puzzle. It preceded and most likely facilited the neo-strategic school of multi-phase logic. The main rule is the model mate: the all mating fields are attacked or blocked just once, and the all white pieces participate in a mate. It's the extra criterion with new combinations suiting #3 or more movers. It gradually produced the unseen quality gaining the international recognition. The miniature #3 with 4 model mates of the leading Bohemian Czech composer *Miroslav Havel* a real surname: Košťál is a classic example. By time the "model mates" options had been exploited, and the neo-strategy provided more combinations.

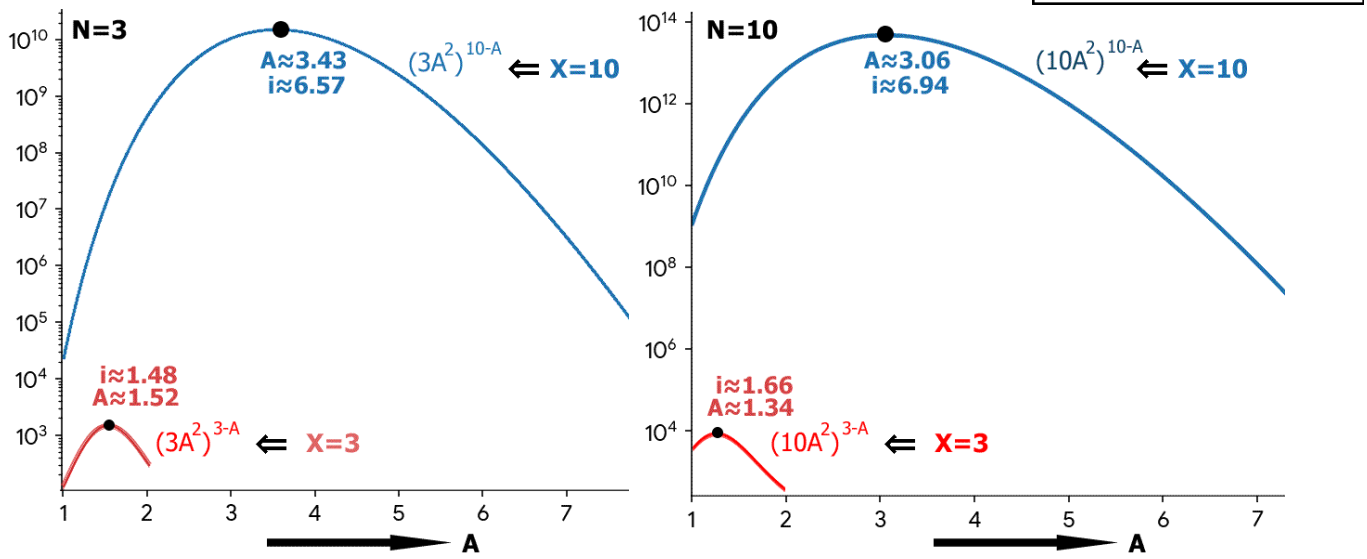
Paul Cézanne

Le panier de pommes, 1893

The Bohemian school is the milestone in the chess composition comparable to the *Cézanne*'s 1879-1880 post-Impressionism of distinct viewpoints recalling "model mates". The multiviews in eg. "The basket of apples" is unreal as the "model mates" are unreal in a classic puzzle. The prestigious exhibition "Paris Salon" was rejecting *Cézanne*'s works. Eg. "Portrait d'Achille Emperaire" 1870 was refused due to unreal incorrect perspective and anatomy. The realistic depiction remains valid criterion, as the difficulty to solve remains valid for a puzzle. But *Cézanne* depicted reality, his apples look like apples. He enhanced, not replaced, the criterion: the multiviews of the "same" reality. The 1st criterion to "depict reality" is a copy "**A**" of the real "**A**" multiplying "**A**": $\mathbf{A^2}$. The more real depiction, the higher **A**. *Cézanne*'s copies of: apples $\mathbf{A_1}$, bottle $\mathbf{A_2}$, table $\mathbf{A_3}$, basket $\mathbf{A_4}$, plate $\mathbf{A_5}$, tablecloth $\mathbf{A_6}$.. are realistic enough, and the multiviews add the intricacy multiplying the copies: $\mathbf{(\Sigma^{N}_{n=1} A_n^2)^i}$. The intricacy of the multiviews exceeds the "copy of the reality" to define the modern art having the higher value lower probability than classic art better imitating copying the reality (with distortions to increase the impact) as *Michelangelo*'s David 1501. The classic puzzles maximize the difficulty to solve: the trickier more obfuscated solution, the better puzzle. Americans Sam Loyd 1841-1911 and William Shinkman 1847-1933, born in Bohemia mastered the surprising and paradoxical keys. The composition's various variants lead to the mates $\mathbf{M_1+M_2+..M_n = \Sigma^{N}_{n=1} M_n}$. The less likely average **M**, the higher composition's value. The Bohemian school maximizes the number of "model mates" - the extra criterion altering the composition's structure to higher use of the pieces. The set of "all model mates" $\mathbf{\Sigma\Theta}$ is a tiny fraction of the set of "all mates" $\mathbf{\Sigma M}$, so the model mates' compositions: $\mathbf{\Sigma^{N}_{n=1} \Theta_n}$, are rarer on average. Some unusual variants of mates $\mathbf{M_x}$ can be rarer than the model mates $\mathbf{\Theta_x}$: $\mathbf{M_x > \Theta_x}$, but the other mates $\mathbf{M_{y/z..}}$ are likely more often than the model mates $\mathbf{\Theta_{y/z..}}$: $\mathbf{M_y < \Theta_y}$, $\mathbf{M_z < \Theta_z}$.. The composition sums its all mates, so it's far likelier the model mates' compositions are rarer uniquer than classic puzzles. So the modernity isn't a mere deviation, it **adds a criterion** multiplying the options. The deviation destroying "**copy of reality**" degrades the art to randomness. To realistically copy the reality is the 1st inevitable criterion. In 1958, *John Gurdon* cloned (=**copied**) a frog from the somatic cells. In 1903, *Wright*'s brothers studied birds to grasp lift and balance (=**copy**) to make the 1st airplane. Da Vinci 1452-1519 invented techniques eg. *Sfumato* to mimic (=**copy**) human vision.

To bend perspective in multiviews *Cézanne* simplified objects **A**↓ to geometrical shapes that inspired *G. Braque* 1882-1963 & *P. Picasso* 1881-1973 to extend the idea in Cubism 1907. The cubistic "**copies of reality**" further simplify objects **A**↓↓ to deepen multiviews **i**↑. So there is a tradeoff between quality of the realistic copy **A**, and the intricacy **i**, when $\mathbf{A}_{realism} > \mathbf{A}_{multiviews} > \mathbf{A}_{cubism}$, while $\mathbf{i}_{realism} < \mathbf{i}_{multiviews} < \mathbf{i}_{cubism}$, where **A**+**i**=**X**. The intricacy **i** exponentionally increases the value, but decreases the quality of **A**. The graphs below illustrate it for various values of **N** and **X**. Both growing number of **A** ⇒ **N**↑ and options **X**↑, increase the intricacy **i**↑ with respect to **A** to reach the maximum.

$$\left(\sum_{n=1}^{N} A_n^2\right)^i, A + i = X$$



$(\Sigma^{N}_{n=1} A_n^2)^i$ can assess seemingly unrelated fields by various values of **A**, **N**, **X**, **i**. The realism copies excerpts of the reality to maximize **A**↑ and **N**↑ in $\mathbf{N*A^2}$. It's the 1st inevitable criterion, so every field includes the realism's parameters: **A**, **N**. Mannerism 16th C intentionally changed some realistic attributes of the realism **N*****A**, as the proloned neck of *Parmigianino*'s "Madonna dal collo lungo" 1534-1540. It's like adding an intricacy to the formula: $\mathbf{N*A_m*i}$, subject to $\mathbf{A_m}$+**i**=**X**. *Arcimboldo* promoted Mannerism using objects creating other object as "Librarian" 1566 from books, but book in itself has no multiuses. The *Cézanne*'s multiviews elevated the intricacy exponentionally $\mathbf{(N*A_v^2)^i}$, as it reuses the "same" in itself (eg. the same apple in distinct views).

The cloning of the organism isn't mere copying the image, but far complex DNA **R** ≫ **A**, with more elements $\mathbf{N_r}$: $\mathbf{N_r} \gg \mathbf{N}$. The 1st film "Roundhay Garden Scene" 1888 by *Louis Le Prince*, was a silent 2.11 sec series of B&W pictures. Pics' sequence exceeds the quantitative extension of the painting, it creates a new quality: "motion" copying the dynamic reality. The chess is zero-sum game. The check-mate **M** is an abstract convention copying the battle. The mate **M** is a tiny subset of the all moves, while the mate $\mathbf{M_1}$ in the chess composition is a tiny subset of the mates **M**. The chess problem deviates to overcome the zero-sum game, to meet extra criterion as Mate in 2. Let's compare eg. 3 real units **3A**, whose qualities lower by 20% in Multiviews and by 60% in Cubism. Evaluation is for *Realism* $\mathbf{3A^2}$, *Multiviews* $\mathbf{(1.92*A^2)^i}$ $=(3*(0.8*A)^2)^i$, *Cubism* $\mathbf{(1.08*A^2)^i}$ $=(3*(0.6*A)^2)^i$.

Realism	BC	$\mathbf{N*A^2}$	$A > 1$
Chess	7C	$\mathbf{N*M}$	
Chess problem	9C	$\mathbf{i*N*M_1}$	$M_1 > M, i>1$
Mannerism	16C	$\mathbf{i*N*A_m^2}$	$A > A_m, i>1$
Model mates	1869	$\mathbf{(N*M_2)^i}$	$M_2 > M_1, i<2, M_2=\Theta$
Film	1888	$\mathbf{N_f*A_f^2}$	$N_f > N, A_f > A$
Multi-views	1893	$\mathbf{(N*A_v^2)^i}$	$A > A_v, i \in (1,2)$
Cubism	1907	$\mathbf{(N*A_c^2)^i}$	$A_v > A_c, i \in (1,2)$
Neo-strategy	1927	$\mathbf{(N*M_1)^i}$	$i \in \{2,3,4..\}$
Cloning	1958	$\mathbf{N_r*R^2}$	$R \gg A, N_r \gg N$

For **A=2**, to have the same values, the intricacy in Multiviews is: **i≈1.2**, in Cubism: **i≈1.7**. The chess problem has higher value than chess, as it's much harder to construct the valid problem than "play chess". That's why the PC algorithms beat the human in chess, but can't so far compose the quality chess compositions. The most valued chess plays are with unexpected combinations, the closest to chess problems / studies. Classic chess problems minimize the probability of the keys / check-mates $\mathbf{M_1}$, *Bohemian school* maximizes the number of the model mates $\mathbf{M_2}$, and neo-strategy maximizes the changes of the mates $\mathbf{M_1}$ and related functionalities in the multi-phases. Unlike *Cézanne*'s multiviews (a bit reducing the quality of **A** =copied unit due to reuse in the multiviews), the "model mates" increase the quality of the mate $\mathbf{M_2}$ that's rarer than "ordinary" mate $\mathbf{M_1}$. But the (very) unexpected key can make the "ordinary" mate $\mathbf{M_x}$ rarer than the "model mate" $\mathbf{M_2}$: $\mathbf{M_x} > \mathbf{M_2}$. So in the chess problem assessment: $\mathbf{i*N*M_1}$, the intricacy **i** maximizes improbable combinations to increase the value of the mate: $\mathbf{i*M_1}=\mathbf{M_x}$, where $\mathbf{M_x} > \mathbf{M_1}$. The

better problem, the higher difference $\Delta(M_x-M_1)\uparrow$. The "model mate" M_2 is on average rarer than the ordinary mate M_1, but can be less rare than the problem with the (very) hidden key: M_x: $M_x>M_2>M_1$. Nevertheless the model mates's value surges with the intricacy **i**: $(N*M_2)^i$ $i\in(1,2)$. Because the model mates aren't the same $\Rightarrow$ $i<2$. In the neo-strategy the intricacy **i** exponentionally increases the value of the mates: $(N*M_1)^i$, where $i \in \{2,3,4..\}$, as the changed mates M_1 are the same in reciprocal / cyclic shifts. For eg. a problem with 2 distinct mates, with $M_x=2$, $M_2=1.5$, $M_1=1$, the value of the "classic puzzle" is: $2*M_x=4$, "model mates": 3^i, "reciprocal-change": 2^i. For the same value **4**, the intricacy **i** in "model mates" is: $i\approx1.26$, "reciprocal change": $i=2$. In fact, the keys of the classic problems aren't usually 2 times rarer than in compositions with "model-mates" / "neo-strategy", which are also often tricky to solve, as they contain various variants and tries. The presented calculations are illustrative, more exact calibration is possible, eg. keys' probabilities can be estimated from structural options different in classic, "model mates" and neo-strategic compositions or empirically by the time in which the solvers find the keys in classic, "model-mates" and "neo-strategic" problems.

Invention, Assessment, Price

Invention is a devation of a form or content with added intricacy **i**↑, keeping the copy imitation of the reality $\approx$**A**. It's within the field as Cubism, Neo-strategy, Quantum mechanics.. or it's a new field based on the new technology as photography, film, PC, internet.. It decreases probabilty to increase uniqueness, independent of time and space. *Gaugin*'s "Tahiti girls" 1890 seemed 'unique' in Paris as Paris seemed 'unique' in Tahiti, but the real art is unique in Paris as well as Tahiti.. So it has value by the way *Gaugin* painted it, not because of Tahiti.. The deviation in itself reduces the value - if it devalues the imitation of reality **A**↓ without sufficient compensation $\approx$**i**. Deviation adds options **o** to imitation **A**, as *Parmigianino* prolonged the neck in the 1st manneristic painting 1535. It's realistic ($\approx$**A**) only intentionally extended (=**o**). But all things (=**N**) can be longer: **N*o**, so the prolongation is uniquer only temporarily: **U=A*(1+time*(o-1))/N*o** time $\in$ (0,1). *H. Bosh* 16C deviates by various proportions + unreal combinations: $N*o_1*o_2*$.. Impressionism 19C deviates by blurring, *Warhol*'s "pop-art" 20C by colors..

So the chronology of the invention is the 1st quality criterion: the earlier, the higher value on average (=not always). It indirectly includes the quality, when to be 1st indicates the distinction from the previous ideas, which is in itself improbable unique. But it's insufficient due to possible multi-discoveries, and mainly due to the other criteria. The greater invention, the more new options, and as was shown the ideas multiplying the existing options as *Cézanne*, Neo-Strategy.. create more options than enhancing or changing parameters: $\approx A^i > i*A$.

The priciest paintings 2025 ranks *Cézanne*'s "The Card Players" 1892 the 3rd ($250M in 2011), *Picasso*'s "Les Femmes d'Alger" 1954 the 9th (179M in 2015), and *da Vinci*'s "Salvator Mundi" 1500 the 1st ($450M in 2017). The most highly insured painting is "realistic" *da Vinci*'s "Mona Lisa" 1503-1519 ($100M in 1962 to be 1.1B in 2025). Although the price isn't 100% objective, already (and not only) as it's changeable and not finite, and many famous paintings aren't for sale, it shows that the new idea (eg. multiview) isn't automatically more valued or better than the before idea (eg. realistic depiction). Surely, it depends on the quality of the concrete painting when eg. a substandard "cubistic" work can have lower value than a quality realistic work. The real value resides in the quality difference Δ**i** in comparison to other works. Eg. "Horses" 20,000 BC of the *cave man*, could be the bigger difference than eg. *Picasso*'s "Le Rêve" 1932.. But how likely it is? The invention enforces certain quality eg. the "neo-strategy" compositions have various qualities, but a substandard "neo-strategy" composition is improbable - like a successful somersault needs a certain style. The higher intricacy **n**↑ of n-jump as eg. a quadruple Axel, the likelier the flaw is, but paradoxically for successful n-jumps: the higher intricacy **n**↑, the probably better style - because any flaw would likely destroy the intricate jump. So the invention to be the invention is flawless. And the more inventions the art contains, the higher value uniqueness and more probable it's worthier than arts with less inventions. As eg. a space rocket is uniquer than a car. The assessment of the invention can confuse 3 partially independent aspects:

- sophistication: a simple wheel versus a advanced space rocket..
- significance: the greatness of the change from the previous inventions..
- price or market value..

Paul Gauguin

Women of Tahiti, 1891

F. Parmigianino

La Madona del cuello largo, 1535

Andy Warhol

Mao, 1972

A cave man

Horses, 20,000 BC

The price imperfectly reflects only certain qualitative aspects. Why pioneering compositions of *G. Cristoffanini* or *A. Mari* have practically no market value, while Picasso's works are sold for millions? Or why footballers earn on average far more than gymnasts or figure skaters, while the gymnastics is far more intricate than a football? Or why

chess or GO players earn far less than tennis or golf players, but far more than chess composers? It seems due to "market size" with a lot more footballers than figure skaters. But, it's not a full explanation as eg. "World Athletics" register 70K active competitors: c. 7x than 9.5K professional golfers, while in 2006, the top golfer *T. Woods* had $**1.3B** - 14.4x than the richest athlete 100m recorder *U. Bolt*'s $**90M**. *Forbes* 2026 lists top annual on-field earnings for fields eg: *C. Ronaldo* **$235M** soccer, *C. Alvarez* **$160M** box, *J. Rahm* **$97M** golf, *M. Parsons* **$83.4M** American football, *M. Verstappen* **$78M** Formula I, *K. Tucker* **$68M** baseball, *S. Curry* **$59.7M** basketball, *C. Alcaraz* **$23.5M** tennis. Maximal annual income for top chess or GO players is $**1-1.4M**, for top figure skaters $**700K** and classic gymnasts $**100K**.. The simpler zero-sum games earn inadequately more than sophisticated fields as gymnastics.
The short answer could be, it's due to the popularity determining ad revenue. And eg. football is accessible and easy to play unlike jumping saults / pirouettes, so logically there is more public attention and so more money. Also the evalution of simple zero-sum games is easy as number of goals, while gymnastics's score assessment reduces the mass interest. It's true in any society, although a capitalism market economy amplifies the role of a profit, so the fields with less people can disappear or not arise at all. Technology or development influences the interest too, when a new field as PC / internet overshadows the previous art forms. Eg. a writer / painter had a higher status in 20C or before than 21C digital age.
The invention is often an idea of one or a few persons, without a guarantee of socio-economic success. It's more improbable than probable that the time of thinking or "innovating" would pay off in terms of a commercial success. Eg. *George Boole* had no profit from his brilliant algebra 1847 - a math foundation of computer science. Also the invention can lead to a degradation as imprisonment of *G. Galilei* 1633. So thinking in itself has a value for inventors, regardless of (expectation of) a profit. It's true regardless of social status, eg. *Cézanne* himself was from a wealthy family and very likely he would gain more pursuing his father's banking career, than the innovative painting, which was a dispute with his father. The renowned philosopher *Heraclitus* 500BC gave up his privileges of aristocratic family to focus on thinking.. Or a lawyer *Pierre de Fermat* 17C born wealthy to devote his leisure to his pioneering math discoveries. It contradicts the classic economics where people maximize the profit and leisure (=no effort)..
To return back to IT frameworks or languages, they are often product of a monopolization, which can seem inevitable. As a common language, you need to learn to communicate. The IT frameworks can be always different or better. Eg. the SQL logic helped unified databases, but it's quite inneficient fragmenting the data to costly rejoin them. And it's very unlikely that innovators as *Fermat*, *Cézanne*, *Newton*, *Boole*.. wouldn't be exploring the better logic / solution. However it would be unpragmatic as job market controlled by IT oligarchy, requires the knowledge of their frameworks regardless of the efficiency. In other words, the monopolized market forms what's likely profitable (learn a framework) and what's not (beyond the framework). As a result, as mentioned before, the regular IT failures are more often as expected. CAST research 2025 estimated "61 billion workdays" needed in software development time to "pay off" the technical debt over the past four decades.., mostly in: USA, Italy, France, India, Spain, Austria, Netherlands.. "Open-source" China isn't in the ladder.. And it's likely reason Chinese "open source" AI DeepSeek overcame American "proprietary" OpenAI..

The order of maximal annual income divided by a number of professionals in field **max(P)/N** 2026 is: $**3.6M** "Formula I" 22 drivers, $**10.2K** 9.5K golfers, $**8.1K** 2.9K tennis players, $**8K** 20K boxers, $**4.7K** 150 figure skaters $**1.8K** 129K footballers, $**1K** 1.4K chess players, $**0.6K** 1-2.5K GO players, $**40** 2-3K classic gymnasts. More relevant can be a number of viewers determining the ad revenue, and average / median income instead of the maximum. It's hard to concentrate the mass attention for figure skaters than the football match. overrates and underrates or discriminatory - burgous or capitalistic society Socialism with far less differences among the sports or fields.

Clearly, any adjustments or metrics considered, some significant differences remain, unexplainable by "market".
sd

as the gymnasts make also far less than tennis or golf players. It's like movie is uniquer than painting, as it includes more inventions.
In general, the price or market value isn't very related to objective assessment of the invention. Eg. while inventions in painting as Alberto Mari or - or in Mathematics - far less money or it's free. Science is a subset of the art
The assessment of the invention or art can be mixed the market value and the artistic contribution.
Karol Mlynka, art nuevou
Film - price... budget
Chess composition with model mates without "Mona Lisa" is the realistic painting **A** with no extra criterion: **i=0**..
Chess is diverse as light athletics long / high / vault jump, javelin / discus / hammer / globe throw, hurdles / runs.. etc, while *Go* is more monotonous as a marathon.
The same number of iterations create more options if they are distributed in more series.

Quality and time, $Q_{uality}(o,i,t) = \Sigma_t^T O_i/I_i$

Quality of the runs depends solely on time: the faster, the better. A marathon takes 2+ hours 35-55,000 steps, 100m sprint 10+ seconds 40-60 steps. A run combined with other definitions eg. a jump, needs other criterion. The high jump takes a few seconds 8-12 steps but it doesn't indicate the achieved height. The acrobatics adds eg. rotation, when neither time nor jump's length can assess the quality. Running is basic inevitable or **horizontal** iteration, and other definitions add **vertical** iterations making the evaluation of the quality more intricate. Eg. assessment of the gymnastics's floor 1+ minute performance covers: "a routine's difficulty D-score & execution E-score, focusing on tumbling precision, dance elements, artistry, & technical skill"..

The distributions of iterations create options defining the activity. Each real distribution has certain options quality. Eg. figure skating in 1-2 minutes can show variously organized harder or simpler jumps. Or #2 with 3 basic iterations can show very intricate as well as trivial schemes. Basic formula is: **Q**uality(o,i) = **o/i**, i = iterations, o = options. The speed is the 1st criterion. The higher speed, the more options per iteration, the higher quality. The world runs' records show the average speed of marathon: 21.2 km/h, of 100m: 37.6 km/h. The both records are the top performances, so have approximately the same quality. The marathon is slower as it lasts much longer. Like chess end-game studies are longer =more moves / iterations with less intricate =slower schemes than in #2, #3. So the Quality sums the qualities of the entire active interval: $\mathbf{Q}(o,i,t) = \mathbf{\Sigma}_t^T \mathbf{O_i/I_i}$. And so the lower qualities can have the same or higher overall quality, if they last long enough.

Meters	Record	km/h	
50	5.56s	**32.4**	1996 D.Bailey
60	6.34s	**34.1**	2018 C.Coleman
100	9.58s	**37,6**	2009 U.Bolt
200	19.19s	**37,5**	2009 U.Bolt
400	43.03s	**33.5**	2016 W.V.Niekerk
800	1:40.91	**28.5**	2012 D.Rudisha
1,500	3:26	**26.2**	1998 H.E.Guerrouj
3,000	7:17	**24.7**	2024 J.Ingebrigtsen
5,000	12:35	**23.8**	2020 J.Cheptegei
10,000	26:11	**22.1**	2020 J.Cheptegei
42,195	1:59:30	**21.2**	2026 S.K.Sawe

Eg. 100m's speed is c. 1.0027 ≈37.6/37.5 higher than 200m's speed. So, 2x more iterations =200m/100m can be ≈0.3% slower per 100m to have the same quality as the 100m's speed. For a marathon it would be 1.27x slower = (42,195/100)*0.003 ie. 29.7 km/h, but the record is only 21.2 km/h - 1.77 slower ≈0.4% per 100m. So, the average quality as speed can fall increasingly to keep the same quality if the **t**ime sufficiently increases. 50m or 60m has lower speed than longer 100m or 200m, which isn't contradictory because the acceleration from the start 0 km/h takes time.

Iterations, Series, Options: $\mathbf{\Pi}_i^N \mathbf{I}_i$, $\mathbf{\Sigma(N-i)*i!}$

$$\mathbf{I}_{\text{terations}} = \sum_{i=1}^{N} \mathbf{I}_i \qquad \mathbf{O}_{\text{ptions}} = \prod_{i=1}^{N} \mathbf{S}_{\text{eries}}(\mathbf{I}_i)$$

Options result from {1, .. , N} Series with {1, .. , N} iterations. Series can combine S(N) absorbs all iterations to coincide with options eg. run: S(N) = O(S). Any independent iteration eg. jump multiplies options. Eg. the high jump flop has c. 4 types of iterations: 1) run, 2) takeoff, 3) flight 4) clearing, ie. $\mathbf{O}_{\text{ptions}} = \mathbf{\Pi}_i^4 \mathbf{I}_i$. Or the paul vault has c. 8 types: 1) run, 2) pole carry, 3) plant, 4) takeoff, 5) swing-up, 6) rock back, 7) turn, 8) clearance, ie. $\mathbf{O} = \mathbf{\Pi}_i^8 \mathbf{I}_i$..

N iterations can be distributed in 1 to N series i ∈ {0, 1, 2, .., N-1}.
The more iterations, the more posible distributions. N iterations can be eg. in:

- 1 series ⇒ Options = N
- 2 same series I_0=N/2, I_1=N/2 ⇒ O=(N/2)*(N/2)=$N^2/4$
- (N/2) series with 2-iterations ⇒ O=$2^{N/2}$
- other combinations 1 to N series with {1, .., N-1} iterations: ..

	O=N	O=$N^2/4$	O=$2^{N/2}$
N=2	2	1	2
N=6	6	9	8
N=10	10	25	32

The possible distributions of series increase with iterations: N ↑ ⇒ D ↑. $\mathbf{S_{(N)}=\Sigma_{i=0}\ (N-i)*i!}$ N=iterations, i=sub-iteration.
The more sub-iterations **i** and the overall iterations **N**, the more options. 10 iterations N=10 in 1 sub-iteration i=0 give 10*0!=10 options, in 5 sub-iterations i=4 6*4!=144 options. It includes also 0 iteration in sub-iteration. For the moment it's sufficient for our analysis.
Mate in **N** has 2*N-1 N*white + (N-1)*black moves basic I_0 iterations. For #2 = 3, #3 = 5.. The 3 iterations in #2 with the iterations of pieces' definitions, can produce a trivial scheme or an intricate cycle. Although #3, #4, #5.. have more basic iterations, the cyclic schemes are harder and since #4, almost impossible. Like a gymnastic performance is too intensive to be long. The chess endgame studies have on average more moves, but fewer pieces c. 3-7 limiting the overall iterations. There are also the studies with more pieces, but with lower usage to limit the overall iterations. So, the more basic iterations, the less extra definitions possible.

Options =quality per unit of time

Meters	Record	km/h	N	per meter	height	
60	7.27s	**29.11**	5	12 8.3%	1.067	2024 G.Holloway

To run is the simplest without other options. Hurdles run+jumps have more options per unit of time, with many variations: height, density, 1st

110	12.80s	**30.95**	10	11	9.1%	1.067	2012	A.Merritt
400	45.94s	**31.34**	10	40	2.5%	0.914	2021	K.Warholm

hurdle's position.. The 110m's 1.2% lower average speed than of 400m, is due to hurdles' 3.64 higher density and 1.167 higher height.

The quality of the records is c. same, so 400m hurdles have less overall options per unit of time than 110m hurdles with less horizontal but multiplied by vertical qualitatively different harder options as to jump is harder than to run. Eg. jumping 10m is harder and takes more time than 10m's run. So the shorter time can have more options than something taking longer. Eg. 2 minutes gymnastic performance Considering the hurdles' heights, for 110m: 10*1.067m=10.67m, for 400m: 10*0.914m=9.14m. The higher density of the hurdles per meter, the more iterations per unit of time.

The cyclic changes in orthodox **Mate in 3** until 1999 +2000-2009

Name	**Scheme**	**Year**	**Author**	**Q=(n*$\bar{q}$)^{I}**	**Number**
Reciprocal	AB-BA	1951	R. Matthews	**4*$\bar{q}^{2}$**	**253** 196 **+57**
Lacny	ABC-BCA	1961	V. Rudenko	**9*$\bar{q}^{2}$**	**60** 46 **+14**
Kiss (key Lacny)	ABC-BCA	1971	V. Kovalenko	**9*$\bar{k}_{2}^{2}$**	**7** 5 **+2**
cyclic Zagoruiko	AB-BC-CA	1973	V. Rudenko	**8*$\bar{q}^{3}$**	**10** 9 **+3**
complete Lacny	ABC-BCA-CAB	1974	M. Velimirovic'	**27*$\bar{q}^{3}$**	**3** 3 **+0**
Shedey (threat Lacny)	ABC-BCA	1976	V. Rudenko	**9*$\bar{t}_{2}^{2}$**	**33** 21 **+12**
4x Lacny	ABCD-BCDA	1978	A. Lobusov J. Retter	**16*$\bar{q}^{2}$**	**4** 4 **+0**
Ukrainian	AB-BC-CA	1988	A. Zidek	**8*$\bar{t}_{1}^{3}$**	**0 / 18** 0 / 15 variations **+3**
Djurašević	ABC-BCA	1988	B. Djurašević	**9*$\bar{h}_{1}^{2}$**	**18** 13 **+5**
4x Shedey	ABCD-BCDA	1990	P. Gvozdják +Ľ. Lehen	**16*$\bar{t}_{3}^{2}$**	**3** 2 **+1**