Generalized Twin Goldbach Primes

Oscar Riveros

Abstract

Clojure & JSR-331 - Puzzles is a set of problems of finite CONSTRAINT LOGIC PROGRAMMING of FINITE DOMAINS, in this document are specifically addressed in Clojure & JSR-331 API The Java Constraint Programming. Itself is a personal investigation, non-profit, is only shared to the public for what it is, a personal study of the issue being raised.

Keywords: Clojure, JSR-331, Primes, Goldbach, Twin

$February 25, 2013$

Definition 1. We say that $2n$ is a *Generalized Twin-Goldbach* number if it can be written by a sum of two primes, say p and q but also between the latter must be the following relationship $q = 2k + p$, where $2k$ (an even number) is the coefficient $2k - Twin$.

Theorem 1. *Let* t_i^k, t_j^l *two* Generalized Twin-Goldbach numbers of the families k and l respectively *then* $t_i^k = p_i + q_i = p_i + p_i + k = 2p_i + k$ *and* $t_j^l = 2p_j + l$ *then* $t_j^l - t_i^k = 2(p_j - p_i) + k + l$ *for all* i, j, l, k.

Problem 1. Create an algorithm that computes all *Generalized Twin-Goldbach* numbers in an interval, and also let it record and $2k - Twin$ coefficient $2k$ v/s *Generalized Twin-Goldbach* number.

Algorithm 1.

```
(ns cpwp.twin-goldbach
  (:use [cpwp.core])
  (:use [clojure.java.io])
  (:import [javax.constraints
            Problem
            ProblemFactory
            Var]))

(def problem nil)
(def top  1000)
(def twin 2)
(def num-solutions 0)

(defn prime?
  [max]
  (not-any? zero? (map #(rem max %) (range 2 max))))

(defn get-primes-domain
  [max]
  (let [primes (cons 2 (for [x (range 3 max 2) :when (prime? x)] x))]
```

```
  (int-array primes)))

(defn make-primes-list
  [prefix l max]
  (let [?list '()]
  (for [i (range l)]
    (cond ?list (.variable problem (str prefix i) (get-primes-domain max))))))

(defn solution-twing-goldbach
  []
  (let [primes (get-primes-domain top)
        length 2
        evens  (int-array (range 2 top 2))
        n      (.variable problem "n" evens)
        q      (make-primes-list "q" length top)]
    (.postAllDifferent problem (into [] (concat [n] q)))
    (.post problem (into-array Integer/TYPE (repeat length 1)) (into-array Var q) "=" n)
    (.post problem (.plus (first q) twin) "=" (last q))))

(defn math
  [next-solution]
  (let [p (.getValue next-solution "q0")
        q (.getValue next-solution "q1")
        n (.getValue next-solution "n")]
    (def num-solutions (+ num-solutions 1))
    (with-open [wrtr (clojure.java.io/writer
                       (str "/Google Drive/tmp/Goldbach/goldbach-twin-" twin ".txt")
                       :append true)]
      (.write wrtr (str n "," p "," q "\n")))))

(doseq [i (range 2 top 2)]
  (def twin i)
  (def problem (ProblemFactory/newProblem (str "Goldbach's Conjecture: t=" twin)))
  (def num-solutions 0)
  (solve-math problem solution-twing-goldbach math)
  (with-open [wrtr (clojure.java.io/writer
                       (str "/Google Drive/tmp/Goldbach/goldbach-twin.txt")
                       :append true)]
      (.write wrtr (str twin "," num-solutions "\n"))))
```

Note 1. In the repository project, there is a zip file with$\sim$ 1000 files with all solutions. (https://github.com/maxtuno/Clojure—JSR-331—Puzzles)

Conclusion 1.

2k Twin	Nº of Generalized Twin–Goldbach number (1000)
2	24
4	26
6	46
8	24
10	32
12	47
14	28
16	24
18	43
20	31

22	25
24	46
26	25
28	25
30	59
32	22
34	26
36	47
38	23
40	31
42	52
44	24
46	23
48	43
50	28
52	24
54	41
56	28
58	19
60	56
62	20
64	21
66	48
68	21
70	33
72	39
74	21
76	23
78	41
80	26
82	23
84	47
86	21
88	21
90	53
92	21
94	22
96	38
98	24
100	24
102	40
104	23
106	19
108	37
110	25
112	22

114	37
116	20
118	21
120	49
122	18
124	21
126	45
128	19
130	24
132	39
134	19
136	20
138	35
140	28
142	17
144	36
146	20
148	18
150	47
152	19
154	24
156	39
158	18
160	26
162	35
164	16
166	18
168	41
170	24
172	18
174	38
176	20
178	18
180	42
182	21
184	17
186	36
188	19
190	25
192	33
194	19
196	20
198	34
200	20
202	16
204	37

206	17
208	20
210	50
212	14
214	15
216	35
218	18
220	26
222	33
224	20
226	19
228	34
230	22
232	17
234	35
236	15
238	21
240	43
242	14
244	16
246	34
248	16
250	24
252	35
254	17
256	15
258	30
260	24
262	14
264	35
266	22
268	14
270	41
272	15
274	14
276	31
278	14
280	23
282	27
284	17
286	18
288	27
290	21
292	16
294	34
296	17

298	12
300	39
302	13
304	14
306	33
308	20
310	19
312	28
314	15
316	16
318	26
320	18
322	17
324	28
326	18
328	16
330	41
332	14
334	13
336	36
338	14
340	18
342	30
344	15
346	16
348	29
350	24
352	14
354	24
356	16
358	13
360	37
362	15
364	17
366	28
368	15
370	20
372	27
374	16
376	17
378	30
380	18
382	14
384	27
386	13
388	12

390	37
392	15
394	13
396	28
398	13
400	15
402	27
404	14
406	17
408	27
410	16
412	14
414	27
416	14
418	15
420	39
422	11
424	13
426	26
428	14
430	16
432	24
434	15
436	15
438	23
440	17
442	13
444	23
446	12
448	14
450	32
452	11
454	13
456	22
458	12
460	17
462	26
464	13
466	10
468	25
470	14
472	9
474	20
476	13
478	11
480	28

482	13
484	12
486	21
488	11
490	16
492	21
494	14
496	13
498	19
500	14
502	11
504	26
506	11
508	8
510	29
512	12
514	9
516	22
518	14
520	14
522	15
524	12
526	10
528	23
530	13
532	9
534	25
536	10
538	11
540	24
542	12
544	11
546	25
548	9
550	14
552	21
554	12
556	10
558	19
560	18
562	10
564	21
566	10
568	8
570	25
572	12

574	11
576	21
578	11
580	14
582	18
584	11
586	8
588	21
590	15
592	7
594	22
596	11
598	8
600	21
602	13
604	10
606	18
608	9
610	10
612	18
614	9
616	10
618	13
620	12
622	8
624	16
626	8
628	6
630	24
632	8
634	9
636	16
638	11
640	10
642	14
644	9
646	9
648	16
650	11
652	5
654	15
656	10
658	8
660	20
662	8
664	8

666	13
668	8
670	10
672	18
674	8
676	5
678	13
680	11
682	7
684	12
686	9
688	7
690	17
692	6
694	5
696	13
698	10
700	8
702	12
704	7
706	5
708	12
710	8
712	6
714	15
716	10
718	5
720	16
722	10
724	6
726	15
728	9
730	9
732	12
734	5
736	7
738	11
740	9
742	5
744	11
746	7
748	6
750	16
752	6
754	8
756	14

758	6
760	7
762	8
764	6
766	7
768	12
770	9
772	3
774	11
776	6
778	7
780	15
782	6
784	8
786	11
788	4
790	7
792	11
794	6
796	5
798	11
800	6
802	5
804	10
806	7
808	6
810	11
812	5
814	5
816	12
818	6
820	7
822	8
824	6
826	5
828	7
830	4
832	3
834	8
836	6
838	3
840	11
842	3
844	5
846	9
848	5

850	6
852	7
854	5
856	3
858	6
860	5
862	2
864	6
866	4
868	4
870	7
872	3
874	4
876	7
878	5
880	3
882	4
884	3
886	1
888	5
890	3
892	2
894	4
896	3
898	3
900	7
902	2
904	4
906	5
908	3
910	4
912	4
914	2
916	4
918	4
920	1
922	3
924	5
926	2
928	2
930	4
932	1
934	4
936	4
938	2
940	2

942	2
944	2
946	1
948	3
950	2
952	1
954	2
956	1
958	2
960	3
962	1
964	3
966	2
968	1
970	2
972	2
974	1
976	1
978	1
980	1
982	0
984	1
986	1
988	1
990	0
992	0
994	0
996	0
998	0