Informatica e pensiero computazionale: una proposta costruttivista per gli insegnanti

Carlo Bellettini[0000−0001−8526−4790], Violetta Lonati[0000−0002−4722−244X], Dario Malchiodi[0000−0002−7574−697X], Mattia Monga[0000−0003−4852−0067], and Anna Morpurgo[0000−0003−0081−914X]

Dipartimento di Informatica, Università degli Studi di Milano, Milan, Italy,
{bellettini, lonati, malchiodi, monga, morpurgo}@di.unimi.it

Sommario L'articolo presenta una proposta formativa che ha per tema la didattica dell'informatica con approccio socio-costruttivista. Tale proposta nasce dall'esperienza sviluppata negli ultimi anni progettando e realizzando *workshop* nelle scuole, e si basa sull'uso di strategie e strumenti costruttivisti per sviluppare il pensiero computazionale e far scoprire l'informatica come disciplina scientifica. Illustriamo gli obiettivi formativi, i contenuti, la metodologia e le attività proposte e descriviamo gli esiti dello svolgimento di due momenti formativi realizzati secondo questa impostazione: un corso rivolto a studenti di laurea magistrale in informatica interessati all'insegnamento e laboratori per insegnanti senza una specifica formazione informatica.

Keywords: Formazione degli insegnanti, didattica dell'informatica, pensiero computazionale, socio-costruttivismo.

1 Introduzione

L'informatica ha un ruolo di prim'ordine nell'impresa umana di accumulazione e diffusione della conoscenza. Ciò crea un legame interessante e peculiare tra informatica e insegnamento: da una parte mettendo in discussione la necessità di fare acquisire procedimenti algoritmici standard (ha ancora senso addestrare a fare le divisioni, vista la disponibilità di dispositivi di facile utilizzo programmati per questo compito?), dall'altro rendendo cruciale che tutti possano sviluppare capacità e competenze specifiche per inventare creativamente nuovi approcci ed escogitare soluzioni per i problemi che si affrontano giornalmente e che, verosimilmente, saranno sempre piú legati all'elaborazione dell'informazione. L'informatica è intrinsecamente riflessione su ciò che si fa e su come lo si fa. Secondo Papert, «pensare a pensare rende ogni bambino un *epistemologo*» [24], e l'informatica racchiude esattamente questo potenziale. Tuttavia, per liberare questo potere è assolutamente necessario evitare di insegnare sterilmente l'informatica tentando semplicemente di *trasferirne* i concetti, o, ancora peggio, di addestrare all'uso di applicazioni software.

Con questo scopo abbiamo progettato vari *workshop* su temi legati all'elaborazione dell'informazione rivolti alle scuole primarie e secondarie, introducendo

l'approccio *algomotorio* [6,5,20,4,19]. Fin dall'inizio abbiamo ritenuto importante impegnarci fortemente in senso costruttivista, al fine di concentrarci sul ruolo centrale dei discenti quali soggetti attivi del processo di apprendimento.

I *workshop* hanno avuto un elevato livello di popolarità, coinvolgendo circa tremila partecipanti nel periodo tra il 2010 e il 2016. Diversi insegnanti hanno deciso di ripetere l'esperienza portando nuove classi o tornando con uno stesso gruppo di studenti per un *workshop* diverso. Tuttavia raramente gli insegnanti hanno deciso di riproporre autonomamente e in prima persona questi *workshop* ai propri studenti, cosí che il bacino di utenza è rimasto confinato in funzione della nostra capacità organizzativa e del tempo a nostra disposizione. Affinché questa iniziativa, che consideriamo di successo, possa realmente diffondersi, è quindi importante formare i singoli docenti in modo che possano applicare l'approccio algomotorio nelle loro classi.

Abbiamo quindi progettato e realizzato alcune occasioni di formazione, rivolte a interlocutori diversi e di formato e lunghezza variabile: eventi indipendenti della durata di poche ore; corsi specifici per la formazione iniziale dei docenti, come il Tirocinio Formativo Attivo; un corso semestrale di "Didattica dell'Informatica" per gli studenti della laurea magistrale in informatica, attivo dal 2015. In tutte queste occasioni, era centrale lo svolgimento dei nostri *workshop*, della durata di circa due ore, seguito da un breve momento di analisi che toccava allo stesso tempo i contenuti informatici, i metodi didattici, gli obiettivi formativi e le premesse metodologiche. L'esito di queste proposte formative è stato simile nei vari contesti: (i) i partecipanti hanno apprezzato l'approccio costruttivista e in molti casi hanno fatto partecipare le loro classi ai nostri *workshop*, ma (ii) non hanno acquisito abbastanza fiducia nella loro capacità di condurre analoghe attività formative in prima persona. Abbiamo attribuito queste insicurezze in parte a una lacunosa preparazione disciplinare (specialmente per gli insegnanti responsabili dell'insegnamento dell'informatica all'interno di altre materie), ma soprattutto all'uso di metodi didattici non convenzionali e incentrati sul discente, che richiedono competenze specifiche relative ad esempio alla gestione dei lavori di gruppo, all'utilizzo di contesti aperti di apprendimento, ecc.: in effetti, del tutto coerentemente con i principi costruttivisti, il semplice *sperimentare* alcune attività costruttiviste non è stato sufficiente per arrivare ad acquisire le competenze necessarie per condurre/erogare interventi didattici di questo tipo.

Questa esperienza ci ha cosí convinti a ripensare le attività di formazione, focalizzandoci su diversi aspetti: (i) l'acquisizione di competenze nell'utilizzo di strategie costruttiviste, (ii) la conoscenza degli elementi fondamentali dell'informatica come disciplina scientifica, (iii) l'applicazione dell'approccio costruttivista ai temi dell'informatica. Il resto dell'articolo è organizzato come segue: in §2 discutiamo di come e perché l'informatica come disciplina scientifica dovrebbe essere insegnata a scuola, mentre in §3 riassumiamo i principi socio-costruttivisti con specifico riferimento all'apprendimento dell'informatica. In §4 descriviamo la proposta formativa, chiarendone gli obiettivi formativi, l'impostazione, le attività proposte; in §5 forniamo un resoconto dello svolgimento di alcuni moduli della proposta formativa e proponiamo una valutazione dei risultati. Il lavoro

termina con alcune osservazioni finali in §6.

2 Competenze trasversali per la scuola

Il dibattito sul perché e come l'informatica dovrebbe essere insegnata a scuola è ricco di interessanti contributi, ad esempio [30,14,11,15,18]. Al di là degli aspetti tecnologici, il contributo culturale piú significativo che l'informatica offre è il cosiddetto "pensiero computazionale", ovvero l'insieme dei processi mentali che mette in atto un informatico nella sua tipica attività di *problem solving*. Si tratta di competenze trasversali, utili e declinabili in tutti gli ambiti disciplinari: formulare i problemi in modo che possano essere risolti in maniera automatica da agenti autonomi, organizzare e analizzare logicamente le informazioni, rappresentarle attraverso modelli e astrazioni, automatizzare lo svolgimento di compiti tramite sequenze di passi ordinati, generalizzare e trasferire processi risolutivi a una grande varietà di situazioni diverse.

Siamo convinti che la ragione principale per insegnare informatica non sia né un aumento, seppur desiderabile, della consapevolezza da parte dei giovani nell'uso della tecnologia, né la preparazione di schiere di futuri informatici. Invece, fare informatica può contribuire allo sviluppo delle competenze citate sopra, richiamate anche nelle "competenze chiave per l'apprendimento permanente" raccomandate dal Parlamento Europeo e dal Consiglio dell'Unione europea [1]. Questa convinzione è supportata anche dalle indicazioni nazionali per il curricolo della scuola del primo ciclo [23]: benché queste citino l'informatica poco e quasi sempre in relazione all'uso degli strumenti informatici, sono numerosi i riferimenti a risultati di apprendimento (non solo in aree affini come la matematica o le scienze, ma anche in aree apparentemente distanti come italiano, storia, musica, ecc.) che possono essere opportunamente associati all'informatica, nel senso che il raggiungimento di tali risultati sarebbe favorito anche da tipiche attività di carattere informatico.

Ad esempio, nella prefazione dell'area disciplinare di Matematica si descrive un modo di ragionare e procedere tipico anche dell'informatica quale scienza che si occupa del trattamento automatico dell'informazione:

> Gradualmente, stimolato dalla guida dell'insegnante e dalla discussione con i pari, l'alunno imparerà ad affrontare con fiducia e determinazione situazioni problematiche, rappresentandole in diversi modi, conducendo le esplorazioni opportune, dedicando il tempo necessario alla precisa individuazione di ciò che è noto e di ciò che s'intende trovare, congetturando soluzioni e risultati, individuando possibili strategie risolutive. [...]
>
> L'alunno analizza le situazioni per tradurle in termini matematici, riconosce schemi ricorrenti, stabilisce analogie con modelli noti, sceglie le azioni da compiere e le concatena in modo efficace al fine di produrre una risoluzione del problema. Un'attenzione particolare andrà dedicata allo sviluppo della capacità di esporre e di discutere con i compagni le soluzioni e i procedimenti seguiti.

L'informatica è anche esplicitamente menzionata nelle "Indicazioni nazionali per i licei" [22], per esempio con i seguenti obiettivi formativi:

> Comprendere la valenza metodologica dell'informatica nella formalizzazione e modellizzazione dei processi complessi e nell'individuazione di procedimenti risolutivi.

E per gli studenti del Liceo scientifico delle scienze applicate:

> Il collegamento con le discipline scientifiche, ma anche con la filosofia e l'italiano, deve permettere di riflettere sui fondamenti teorici dell'informatica e delle sue connessioni con la logica, sul modo in cui l'informatica influisce sui metodi delle scienze e delle tecnologie, e su come permette la nascita di nuove scienze.

A fronte di queste considerazioni, i temi dell'informatica che ci sembra significativo proporre nel contesto scolastico si possono riassumere come segue: (i) rappresentazione dell'informazione e organizzazione dei dati, (ii) programmazione informatica, (iii) pensiero algoritmico e strategie per la risoluzione di problemi.

3 Socio-costruttivismo e informatica

Per un'introduzione alle teorie socio-costruttiviste e alla loro applicazione ai contesti didattici, si può fare riferimento al libro [9]; qui riassumiamo gli elementi fondamentali.

Secondo l'epistemologia costruttivista, che ha le sue origini nel pensiero, tra gli altri, di Piaget, la conoscenza si *costruisce* attraverso l'esperienza e la riflessione sull'esperienza stessa; quindi è colui che apprende che crea la propria conoscenza, e questa creazione si basa fortemente su ciò che egli già sa e capisce [25]. In particolare il *costruttivismo radicale* [13] nega l'esistenza di una verità *ontologica* perché la conoscenza, derivando dall'esperienza di colui che apprende, è *unica* per definizione. Di conseguenza, è impossibile trovare una rappresentazione di una realtà indipendente pronta da essere *trasferita* agli studenti. Diventa quindi necessario chiarire la distinzione tra ciò che è conoscenza e ciò che non lo è: il criterio adottato, ispirato alla teoria dell'evoluzione, è se la conoscenza è viabile (dall'inglese "*viable*") ovvero funziona soddisfacentemente nelle esperienze avute fino a questo momento. Il socio-costruttivismo integra anche il principio di Vygotskij secondo cui lo sviluppo della conoscenza è guidato e influenzato dal contesto sociale, e quindi dalle interazioni con gli altri individui e in particolare dal loro uso del linguaggio [32].

Sulla base di queste premesse, un insegnante non può preparare rappresentazioni della realtà "preconfezionate" da trasferire ai suoi allievi. Piuttosto, il suo ruolo è di aiutare la costruzione di conoscenza attraverso stimoli, metodi e strategie che favoriscano l'attivazione del processo di apprendimento. In questo processo, l'attore finale è proprio il discente, che deve quindi modificare le proprie rappresentazioni mentali in modo da adattarle alle nuove esperienze.

Il fatto che la conoscenza debba essere costruita individualmente implica che ogni allievo dovrebbe poter sviluppare un percorso di apprendimento personale, del quale è il principale responsabile, in cui il docente accompagna le sue scoperte incoraggiando riflessioni *metacognitive* riguardo a come la comprensione si

sviluppa. Questo tipicamente significa incoraggiare l'allievo a usare inizialmente tecniche di apprendimento attivo per creare nuova conoscenza (esperimenti, soluzione di problemi concreti) e successivamente a riflettere e discutere riguardo a cosa sta facendo e come la sua conoscenza sta cambiando. Per questo è fondamentale che l'insegnante si assicuri di avere compreso quali sono le convinzioni (anche eventualmente erronee) preesistenti nel discente, per tenerne conto e usarle come punto di partenza nel progettare e condurre le attività.

Tra gli aspetti principali che un insegnante costruttivista dovrebbe considerare, citiamo i seguenti: l'efficacia dell'apprendimento cooperativo [17]; l'esistenza di diversi stili e strategie di apprendimento [12]; l'importanza dell'ascolto attivo [29], del feedback personalizzato [31], della riflessione metacognitiva [10]; l'utilità di progettare e proporre occasioni di apprendimento basate su compiti autentici e realistici [26]; il cambio di paradigma che vede l'insegnante come facilitatore del processo di apprendimento [27] e ricercatore riflessivo [28] che apprende dalla sua riflessione come migliorare la sua qualità professionale.

L'approccio costruttivista è particolarmente efficace nell'apprendimento dell'informatica e nello sviluppo del pensiero computazionale [7,21], in cui la capacità di risolvere problemi (*problem solving*) riveste un ruolo centrale. Una metodologia che applica strategie costruttiviste ad argomenti informatici è l'*algomotricità*. Come suggerisce il nome (un neologismo sincratico che combina le parole *algoritmo* e *motorio*), l'algomotricità sfrutta attività cinestetiche nelle quali gli studenti hanno l'occasione di confrontarsi in maniera informale con uno specifico tema informatico. Questo primo momento operativo è seguito da attività che favoriscono un processo di astrazione nel quale gli studenti arrivano a costruire modelli mentali appropriati al concetto che si sta esplorando. Come ultima fase viene prevista una attività basata sui computer per ricollegarsi a qualcosa che gli studenti riconoscono naturalmente come informatica. Le attività partono infatti sempre "*unplugged*" [2], ma finiscono con un lavoro in cui gli studenti si confrontano con delle applicazioni *software* concepite apposta per esplicitare il legame con le tecnologie informatiche. Ad esempio, la Fig. 1 mostra un'attività algomotoria ("wikipasta" [6,4]), progettata per riflettere sulla codifica di informazione e meta-informazione nei testi formattati: alla fine gli studenti usano un'applicazione in cui possono produrre un testo formattato tramite un linguaggio a marcatori analogo a quello in uso per Wikipedia.

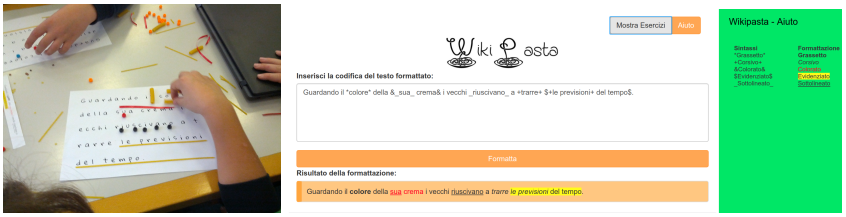

Figura 1. Un'attività algomotoria progettata per riflettere su informazione e meta-informazione; a destra l'applicazione sviluppata per concludere l'esercitazione.

4 Descrizione della proposta formativa

Considerando le indicazioni nazionali e tenendo conto del contesto scolastico e della formazione degli insegnanti [3], presentiamo una proposta formativa che si può articolare diversamente a seconda dei docenti coinvolti, e si compone dei seguenti moduli.

A. L'informatica e il pensiero computazionale come competenze trasversali.
B. Didattica dell'informatica con approccio costruttivista.
C. Didattica per competenze in informatica.
D. Didattica della programmazione.

Il modulo A è pensato per persone senza una formazione specifica nell'ambito; i moduli B e C si occupano dell'insegnamento dell'informatica sia nell'ambito di materie dedicate che in contesti piú generalisti; il modulo D è dedicato all'insegnamento della programmazione in contesti specialistici, da parte di docenti con una formazione specialistica in informatica.

Il costruttivismo non è solo una parte centrale dei contenuti della proposta formativa, ma è anche esso stesso l'approccio didattico usato. Pertanto si prevedono attività di discussione e/o *problem solving*, da svolgere in piccoli gruppi, alternate a momenti di ricapitolazione del lavoro svolto e brevi spiegazioni frontali. Simulazione e compiti autentici sono usati per dare l'occasione ai partecipanti di immaginarsi/calarsi concretamente nel ruolo di docenti. Si propongono schede di riflessione metacognitiva sia come strumenti di autovalutazione durante i moduli che come strumenti di lavoro da utilizzare nella pratica didattica. Per ogni modulo, all'inizio delle attività, vengono esplicitamente dichiarati le premesse metodologiche, gli obiettivi didattici e i criteri di valutazione. Nel seguito, riassumiamo gli obiettivi e i contenuti di ciascun modulo, esemplificandoli con la breve descrizione di alcune delle attività proposte.

A. L'informatica e il pensiero computazionale come competenze trasversali. Questo modulo ha l'obiettivo di presentare l'informatica come disciplina scientifica a persone senza una formazione specifica nell'ambito. Mediante attività di tipo algomotorio, i partecipanti hanno modo di esplorare alcuni temi informatici fondamentali e mettere in pratica le abilità tipiche del pensiero computazionale (vedi §2), facendo allo stesso tempo esperienza diretta, nel ruolo di discenti, di strumenti e strategie costruttiviste (vedi §3. Ad esempio: *feedback* individualizzato, compiti autentici). Il modulo ha inoltre l'obiettivo di motivare l'insegnamento dell'informatica come disciplina che favorisce lo sviluppo di importanti competenze trasversali, evidenziando in particolare i collegamenti possibili con le altre discipline. Le attività proposte sono per lo piú tratte e riadattate dai nostri laboratori didattici [5] o ispirate a quesiti Bebras [8], un'iniziativa divulgativa in cui si propongono piccoli problemi "informatici" a squadre di 4 studenti.

B. Didattica dell'informatica con approccio costruttivista. Il modulo ha l'obiettivo di far conoscere e sperimentare, questa volta dal punto di vista del docente, elementi teorici, strategie, metodi e strumenti della didattica costruttivista,

nonché di mostrare, attraverso un ampio repertorio di attività *unplugged* o al computer, come questi si possano applicare in particolare ai temi fondamentali dell'informatica. L'attenzione in questo caso si concentra sugli aspetti didattici relativi sia al contenuto informatico delle attività che agli aspetti metodologici relativi al ruolo e alle capacità del docente facilitatore (ad esempio: gestione dei lavori di gruppo e del tempo, consegne e *feedback* durante le attività di *problem solving*, ascolto attivo, presidio dei momenti di riflessione metacognitiva, ecc.). Ad esempio, si propone di confrontare una lezione frontale e un breve percorso di didattica attiva relativi allo stesso tema informatico, analizzando per entrambi il materiale didattico (slide e libro di testo *versus* ambiente di apprendimento interattivo e scheda di lavoro) e il ruolo del docente e dei discenti. Oppure si fornisce a un gruppo la descrizione di un'attività didattica e i relativi materiali e si chiede di simularne l'erogazione a un altro gruppo (consegna, gestione del tempo, monitoraggio e accompagnamento), concludendo con una valutazione del lavoro dal punto di vista dei due diversi ruoli di docente e discente. Un'ulteriore attività proposta consiste nell'esaminare materiali prodotti realmente da gruppi di studenti, e di rielaborarli al fine di produrne una sintesi che valorizza i contributi di tutti (la cosiddetta "restituzione").

C. Didattica per competenze in informatica. Il tema del modulo è la progettazione, pianificazione e valutazione di unità di apprendimento di temi informatici, nel paradigma di riferimento della didattica per competenze. Partendo innanzitutto dall'analisi critica di proposte didattiche già esistenti (anche scelte tra quelle viste nei due moduli precedenti), i partecipanti sperimentano e imparano a: definire gli obiettivi formativi e i criteri e gli strumenti di valutazione di tali obiettivi (p.es.: rubrica); suddividere le unità di apprendimento in fasi successive pianificando le varie attività; predisporre i materiali e le consegne; riflettere sul proprio lavoro, rivedendo e adattando il proprio progetto didattico.

D. Didattica della programmazione. Questo modulo è dedicato all'insegnamento della programmazione in contesti specialistici. Si propone l'uso di attività *unplugged* e di ambienti di programmazione visuali per introdurre il tema; si discute inoltre dell'importanza di proporre problemi significativi e di scegliere un linguaggio di programmazione adatto allo sviluppo di progetti concreti e realistici. Inoltre, a partire da esercitazioni pratiche, si affronta il tema delle convinzioni preesistenti dei discenti (in inglese *misconception*) nell'apprendimento della programmazione, ci si concentra sull'importanza del *feedback* e si riflette sul processo di *testing/debugging* come occasione paradigmatica di utilizzo del metodo scientifico.

5 Realizzazione della proposta formativa e valutazione degli esiti

Sulla base di quanto descritto in §4, nel 2017 abbiamo realizzato: (i) una serie di laboratori di 3 ore ciascuno, corrispondenti al modulo A, per insegnanti senza

una formazione specifica di informatica; (ii) una nuova versione del corso "Didattica dell'Informatica" per la laurea magistrale in informatica, composto dai moduli B, C, e D (a ogni modulo sono state dedicate circa 16 ore, suddivise ciascuna in 4 sessioni). Altri momenti formativi (del corso intero o di alcuni moduli soltanto) si svolgeranno nei prossimi mesi nell'ambito del programma CS[4]HS di Google [16].

I laboratori hanno visto una presenza di 28 persone, mentre il corso è stato seguito da 14 studenti (12 con una laurea triennale in informatica e 2 insegnanti di scuola primaria, che non hanno partecipato al modulo D). In entrambi i casi, la proposta ha suscitato interesse e partecipazione attiva; in particolare durante il corso, la qualità delle produzioni degli studenti è stata alta e in costante miglioramento grazie al *feedback* ricevuto.

Ai partecipanti è stato inoltre chiesto di compilare un questionario per rilevare le loro impressioni. Le domande richiedevano risposte su una scala di Likert e alcuni commenti aperti. La domanda "La proposta ha accolto le tue aspettative?" ha ricevuto solo risposte positive; la domanda "Qual è il tuo livello complessivo di soddisfazione?" ha ricevuto solo risposte positive, con circa metà di voti massimi.

In particolare per quanto riguarda il corso, le valutazioni su la scelta degli argomenti, la metodologia seguita, i materiali utilizzati e la competenza dei docenti sono state altrettanto positive, con un'unica risposta debolmente negativa. I partecipanti hanno apprezzato la combinazione di teoria e pratica, l'ampia possibilità di discutere e condividere le proprie idee e la riflessione su aspetti dell'informatica che nei corsi piú tradizionali vengono talvolta dati per scontati (come, ad esempio, il concetto di algoritmo), la presentazione di linguaggi visuali, il confronto tra linguaggi di programmazione nel contesto didattico e l'idea di riflettere su errori comuni e fraintendimenti ricorrenti. Gli aspetti piú critici sono stati la mancanza di tempo, soprattutto per completare le attività di gruppo, la difficoltà di immaginare l'interazione con i discenti e la necessità di maggiore approfondimento di alcuni temi. Inoltre, talvolta le presentazioni di gruppo sono risultate un po' ripetitive.

I docenti che hanno partecipato ai laboratori hanno dichiarato che l'esperienza li ha divertiti, stimolati, interessati, incuriositi, soddisfatti, arricchiti, chiariti, convinti (18 risposte) ma anche, in alcuni casi, confusi o affaticati (2 risposte). Inoltre hanno suggerito vari contesti di applicabilità a scuola delle attività e dei metodi proposti.

6 Conclusioni

Convinti che il modo scientifico di pensare caratteristico dell'informatica sia essenziale per essere creativi in un mondo dove l'elaborazione dell'informazione si intreccia sempre piú profondamente con le nostre vite, alcuni anni fa abbiamo messo a punto alcune attività di sensibilizzazione intese a portare nelle scuole gli aspetti essenziali di questa disciplina, progettandole a partire da principi costruttivisti. Gli insegnanti hanno risposto positivamente ai *workshop* proposti,

ma spesso non si sono sentiti di replicarli autonomamente nelle loro classi. Abbiamo quindi iniziato a proporre seminari di formazione per gli insegnanti per poi sviluppare a partire da questi un insegnamento di "Didattica dell'informatica" per un corso di laurea magistrale che applica l'approccio costruttivista anche a se stesso e infine un corso modulare per insegnanti focalizzato sull'acquisizione di competenze nell'utilizzo di strategie costruttiviste, su aspetti didattici cruciali nell'insegnamento dell'informatica (declinando i contenuti in modo differente per contesti generalisti e specialistici) e sulla formazione relativa alle basi della disciplina. I risultati, sebbene ancora iniziali, sono incoraggianti: i partecipanti alle nostre iniziative hanno dimostrato interesse e soddisfazione, ma, soprattutto hanno finalmente suggerito vari contesti di applicabilità a scuola delle attività e dei metodi proposti, molti dei quali del tutto originali. Speriamo quindi di poter rilevare presto un'efficacia piú ampia dei nostri interventi.

Ringraziamenti

Gli autori ringraziano Google Inc. per l'aiuto fornito tramite il programma CS[4]HS e il Dipartimento di Informatica dell'Università degli Studi di Milano per il supporto delle attività di ALaDDIn LAboratorio di Divulgazione e Didattica dell'INformatica.

Riferimenti bibliografici

1. Raccomandazione del Parlamento europeo e del Consiglio, del 18 dicembre 2006, relativa a competenze chiave per l'apprendimento permanente. http://data.europa.eu/eli/reco/2006/962/oj, Dec. 2006.
2. T. Bell, J. Alexander, I. Freeman, and M. Grimley. CS unplugged: Imparare l'informatica senza alcun computer. https://classic.csunplugged.org/wp-content/uploads/2016/02/csunplugged-it.2015.1.0.pdf, 2015. Ed. italiana a cura di R. Davoli, G. M. Bianco, P. Grossi.
3. C. Bellettini, V. Lonati, D. Malchiodi, M. Monga, A. Morpurgo, and F. Pedersini. *E questo tutti chiamano informatica. l'esperienza dei TFA nelle discipline informatiche*, volume 14 of *Collana Manuali*, chapter La formazione degli insegnanti della classe 42/A — Informatica: l'esperienza dell'Università degli Studi di Milano, pages 53–76. Sapienza Università Editrice, 2015.
4. C. Bellettini, V. Lonati, D. Malchiodi, M. Monga, A. Morpurgo, and M. Torelli. What you see is what you have in mind: constructing mental models for formatted text processing. In *Proceedings of ISSEP 2013*, pages 139–147, 2013.
5. C. Bellettini, V. Lonati, D. Malchiodi, M. Monga, A. Morpurgo, M. Torelli, and L. Zecca. Extracurricular activities for improving the perception of informatics in secondary schools. In *Proceedings of ISSEP 2014*, pages 161–172, 2014.
6. C. Bellettini, M. Monga, V. Lonati, A. Morpurgo, D. Malchiodi, and M. Torelli. Exploring the processing of formatted texts by a kynesthetic approach. In *Proceedings of WiPSCE 2012*, pages 143–144. ACM, 2012.
7. M. Ben-Ari. Constructivism in computer science education. *ACM SIGCSE Bulletin*, volume 8, 1998.

8. A. Calcagni, V. Lonati, D. Malchiodi, M. Monga, and A. Morpurgo. Promoting computational thinking skills: Would you use this Bebras task? In *Informatics in Schools: Focus on Learning Programming - 10th International Conference on Informatics in Schools: Situation, Evolution, and Perspectives, ISSEP 2017, Helsinki, Finland, November 13-15, 2017, Proceedings*, volume 10696 of *Lecture Notes in Computer Science*, pages 102–113. Springer, 2017.
9. A. Carletti and A. Varani, editors. *Didattica costruttivista. Dalle teorie alla pratica in classe.* Erickson, 2005.
10. C. Cornoldi. *Matematica e metacognizione: atteggiamenti metacognitivi e processi di controllo*, volume 43. Edizioni Erickson, 1995.
11. V. Dagienė and S. Sentance. It's computational thinking! Bebras tasks in the curriculum. In *Proceedings of ISSEP 2016*, pages 28–39. Springer, 2016.
12. R. S. Dunn and K. J. Dunn. *Teaching students through their individual learning styles: a practical approach.* Reston Pub. Co., Reston, VA, 1978.
13. E. V. Glasersfeld. Idee costruttiviste. *Riflessioni Sistemiche*, (2):179–181, 2010.
14. O. Hazzan, T. Lapidot, and N. Ragonis. *Guide to Teaching Computer Science: An Activity-Based Approach.* Springer London, 2011.
15. J. Hromkovič and R. Lacher. The computer science way of thinking in human history and consequences for the design of computer science curricula. In *Proceedings of ISSEP 2017*, pages 3–11. Springer, 2017.
16. G. Inc. CS[4]HS. https://www.cs4hs.com/resources-materials.html, 2017.
17. D. W. Johnson and R. T. Johnson. *Learning together and alone: Cooperative, competitive, and individualistic learning.* Prentice-Hall, Inc, 1987.
18. M. Lodi, S. Martini, and E. Nardelli. Abbiamo davvero bisogno del pensiero computazionale? *Mondo digitale*, (72):1–15, Nov. 2017.
19. V. Lonati, D. Malchiodi, M. Monga, and A. Morpurgo. Is coding the way to go? In A. Brodnik and J. Vahrenhold, editors, *ISSEP 2015*, pages 165–174, 2015.
20. V. Lonati, D. Malchiodi, M. Monga, and A. Morpurgo. Nothing to fear but fear itself: introducing recursion in lower secondary schools. In *Proceedings of LATICE 2017*, pages 91–98, 2017.
21. R. Marchignoli and M. Lodi. *EAS e pensiero computazionale.* La Scuola, 2016.
22. Ministero dell'Istruzione, dell'Università e della Ricerca. Indicazioni nazionali per i percorsi liceali, 2010.
23. Ministero dell'Istruzione, dell'Università e della Ricerca. Indicazioni nazionali per il curricolo della scuola dell'infanzia e del primo ciclo d'istruzione, 2012.
24. S. Papert. *Mindstorms: Children, computers, and powerful ideas.* Basic Books, Inc., 1980.
25. J. Piaget and B. Inhelder. *The Psychology of the Child.* Basic Books, 1969.
26. P. C. Rivoltella, A. Garavaglia, S. Ferrari, A. Carenzio, E. Bricchetto, L. Petti, and S. Triacca. Fare didattica con gli EAS. Episodi di apprendimento situato. *DIDATTICA*, pages 5–241, 2013. Ed. La Scuola.
27. C. Rogers and R. E. Farson. Active listening. *Organ. Psyc.*, pages 168–180, 1979.
28. D. A. Schön. *Il professionista riflessivo.* Dedalo, 1993.
29. M. Sclavi. *Arte di ascoltare e mondi possibili. Come si esce dalle cornici di cui siamo parte.* Pearson Italia Spa, 2003.
30. The Royal Society. Shut down or restart? The way forward for computing in UK schools., Jan. 2012.
31. M. Thurlings, M. Vermeulen, T. Bastiaens, and S. Stijnen. Understanding feedback: A learning theory perspective. *Educ. Research Review*, 9:1–15, 2013.
32. L. Vygotsky. *Mind in Society: Development of Higher Psychological Processes.* Cambridge: Harvard University Press, 1978.