TEMA

La computer grafica per una didattica per problemi

Marisa Di Luca[1], Giorgio Bolondi[2], Fiorenza Papale[3]

[1]IIS A. Volta, Pescara, marisadl@libero.it

[2]Università di Bologna, giorgio.bolondi@unibo.it

[3]Dirigente Scolastico IIS A. Volta, Pescara, fiorenza.papale@istruzione

Perché i problemi?

Da diversi anni, per quanto riguarda l'insegnamento della matematica, l'attenzione è sempre più focalizzata sulle attività di processo e di "ragionamento" piuttosto che sul "risolvere esercizi", e "l'applicare formule". Questo ha ridato vigore ad un qualcosa che in matematica si fa da sempre: risolvere problemi.

Ma siamo sicuri che quando chiediamo ai nostri studenti di risolvere un problema in realtà non proponiamo loro un esercizio "travestito" da problema? Partiamo da un esempio: consultando un libro di statistica si può trovare alla voce "problemi" un testo del tipo: "**Data la seguente tabella che rappresenta i risultati di una indagine sul gradimento di alcuni treni, costruire la retta di regressione di X su Y e di Y su X**".

	TIPO DI TRENO			
SODDISFATTO	**EUROSTAR**	**INTERCITY**	**INTERREGIONALI**	**TOTALE**
SI	203	118	178	499
NO	122	167	528	817
TOTALE	**325**	**285**	**706**	**1316**

Che tipo di problema è? Lo studente è costretto ad un esercizio mnemonico per tentare di ricordare le formule per calcolare i coefficienti della retta di regressione, non coinvolge nessuna abilità di pensiero, non è costretto al ragionamento. Questo è un esercizio, non un problema.

Iniziamo quindi da una prima doverosa distinzione: quella tra **esercizio** e **problema**.

Il problema, quello vero, pretende che lo studente si impegni in tutta serie di passi che lo portano alla soluzione e lo costringono alla riflessione critica sul processo seguito.

Bolondi definisce il problema come "*situazione in cui si presenta una domanda a cui bisogna dare una risposta costruendo una strategia risolutiva che utilizza anche strumenti di aritmetica, geometria, logica…*".

Quindi si può affermare che:

- risolvere un esercizio consiste in una attività sempre uguale (a volte cambiano solo i numeri) che non coinvolge in minima parte la creatività dello studente
- risolvere un problema comporta il coinvolgimento di attività creative: lo studente riorganizza abilità consolidate per ideare strategie che non ha mai utilizzato in precedenza.

I problemi, quindi, risultano essere fondamentali nell'insegnamento e nell'apprendimento della matematica; le procedure di soluzione richiamano operazioni quali: definizione, dimostrazione, astrazione, generalizzazione, schematizzazione, rappresentazione che con un semplice esercizio sono solo in parte coinvolte.

Il Problem Posing e Solving

In matematica si parla quasi sempre di *problem solving*, quasi mai di *problem posing*. Vediamo di puntualizzare questi due concetti che sono strettamente legati, interconnessi.

Il *problem posing* (porre problemi) come metodo didattico supera la dicotomia discente–docente e focalizza l'idea che l'insegnante non può essere il depositario del sapere, che la conoscenza non si può trasmettere da un soggetto (il docente) ad un altro (il discente), ma l'apprendimento avviene tramite il dialogo tra i soggetti coinvolti. **Paulo Freire** (pedagogista brasiliano, 1921-1997) è stato lo studioso che ha teorizzato il *problem posing*, secondo il suo pensiero: *l'educazione autentica non viene portata da A per B o da A circa B bensì da A con B*. Come non essere d'accordo! Gli insegnanti (facilitatori in un processo di insegnamento/apprendimento) devono: ascoltare, dialogare e quindi agire.

Il problem posing si può definire come una processo cognitivo che attraverso la formulazione di problemi su affermazioni relative a determinate situazioni/oggetti e sulla successiva "negazione", messa in discussione di dati e proprietà della situazione/oggetto può portare a ipotesi alternative. Siamo di fronte ad un vero e proprio atto creativo.

Le fasi del *problem posing* sono cinque, vediamole:

- Si individua la situazione/oggetto.
- Si costruisce un elenco delle caratteristiche che si ritengono fondamentali a definire la situazione/oggetto.
- Si inizia a negare quanto prima individuato.
- Si pongono domande, problemi nuovi.
- Si risolvono i problemi "costruiti".

Come si può vedere l'ultima fase è di *problem solving*, ma del problema o dei problemi che si sono costruiti partendo da una determinata situazione.

Nella fase 4 può essere un valido strumento il *brainstorming*; c'è la possibilità di lavorare anche sul pensiero divergente, sugli aspetti creativi. Una delle tecniche più coinvolgenti da utilizzare con gli studenti sicuramente è quella del **pensiero rosso** (tutti i componenti del gruppo esprimono le proprie idee anche se palesemente errate o che sembrano sciocche) e del **pensiero verde** (il gruppo seleziona quelle che ritiene idee utili allo svolgimento del compito, fase di validazione); il gruppo si adegua alle decisioni della maggioranza ed esegue.

Il *Problem Solving* ovviamente tende alla ricerca della soluzione ad un problema; tale soluzione non necessariamente è di tipo numerico: problemi di determinazione; ci sono problemi di costruzione quando si "costruisce" un oggetto geometrico; si parla di problemi di dimostrazione

quando si vuol arrivare, ovviamente, alla dimostrazione di un teorema, di una legge etc etc

Inoltre i problemi non si risolvono solo con algoritmi (procedure ben definite che garantiscono un risultato esatto) ma attraverso anche "procedure euristiche" cioè metodi "deboli" applicabili anche quando non è possibile l'uso di algoritmi.

I passi del *problem solving*:

- TRADUZIONE: ogni affermazione contenuta nel testo viene trasformata in una rappresentazione semantica in memoria; analisi linguistica e semantica.
- INTEGRAZIONE: mettere insieme in maniera coerente tutte le differenti parti del problema in una struttura unitaria; importante è la categorizzazione", cioè l'individuazione della categoria generale alla quale il problema può appartenere.
- PIANIFICAZIONE: individuazione della strategia da seguire per trovare la soluzione.
- CALCOLO: individuazione delle operazioni necessarie e il procedimento.

Come si collega il *computational thinking* (pensiero computazionale, CT) con il problem posing e solving? Il CT, teorizzato da **Seymor Papert** (matematico sudafricano naturalizzato americano, 1929), si può definire come un metodo di risoluzione di problemi che utilizza la tecnologia informatica. Vediamo i passi:

- scomposizione del problema in sottoproblemi, più semplici, verifica se i singoli sottoproblemi sono stati già affrontati. La scomposizione di un problema porta alla costruzione di un algoritmo risolutivo; Es: 1) individuare i componenti quando assaggiamo un nuovo piatto, 2) quando diamo indicazioni di una strada.
- definizione degli elementi costitutivi di un oggetto/situazione che daranno la possibilità di costruire **modelli**. Es: 1) studenti che cercano di capire quali comportamenti sono graditi ai loro docenti, stanno identificando un modello.
- individuazione delle informazioni necessarie alla soluzione e generalizzazione per poter "esportare" la soluzione ad altri problemi simili; costruzione della sequenza di passi per la soluzione non del problema specifico, ma di tutti i problemi di una determinata classe (algoritmo).

E' a tutti gli effetti un processo di *problem solving*!

Vediamo un esempio di utilizzo della tecnologia che può essere considerato a tutti gli effetti ottimo per il *problem solving*: la computer grafica.

La **computer grafica**, per le sue caratteristiche, è particolarmente adatta ad una impostazione della didattica in cui l'approccio ***per problemi*** e gli aspetti ***laboratoriali*** risultano essere il fulcro dell'attività stessa. L'utilizzo di un software in questo senso non deve essere confuso con prodotti tecnologici dedicati alla matematica come ad esempio: Derive, GeoGebra, Cabrì. Questi pacchetti sono utilizzati principalmente per "verificare" teoremi, regole, formule, concetti successivamente alla loro discussione in classe, o comunque per l'esplorazione di contenuti già pienamente strutturati e organizzati in altri ambienti di apprendimento. Nel caso della computer grafica lo studente si trova di fronte ad una situazione problematica: rappresentare (in 2D o 3D) oggetti comuni come ad esempio un albero, un bicchiere, un tavolo e deve capire **QUALE** matematica occorre e **COME** utilizzarla. Lo strumento tecnologico (hardware e software) è al tempo stesso ambiente di apprendimento, strumento di mediazione didattica, mediatore semiotico radicalmente differente da quelli tradizionali. Il software di riferimento è POV-Ray (www.povray.org), un prodotto open source, Si tratta di un cosiddetto *ray-tracer*: il programma legge una descrizione testuale di una scena costituita da un certo numero di oggetti geometrici, sorgenti di luce, etc. e ne genera un'immagine fotorealistica con ombre, riflessi, trasparenze, etc. POV-RAy dà l'opportunità di "utilizzare" la matematica in un contesto veramente differente, che favorisce, inoltre, un apprendimento per competenze.

Quando si costruisce con POV-Ray si parla di "**scena**", vediamo di cosa si tratta; una descrizione di scena può contenere le seguenti componenti:

- **#include**: per richiamare i file che contengono dichiarazioni predefinite per i colori, le strutture e altro.
- **camera**: per definire l'angolo visuale, il punto di vista.
- **light_source**: per definire la posizione, il colore della luce.
- **oggetti geometrici e loro proprietà**:
  - in questo caso possiamo avere come figura un ente geometrico di base(sfera, cilindro, piano, …) oppure un oggetto composto con figure geometriche di base con i metodi di **CSG: Geometria Solida Costruttiva** (possibilità di "costruire" solidi complessi partendo da quelli fondamentali utilizzando operazioni che "si ispirano" alle funzioni booleane).
  - **trasformazioni**: per modificare la posizione della figura; per

definire una trasformazione è necessario precisare la scala, per questo si usa la scale<>; possiamo ruotare intorno al centro del sistema di coordinate con: **rotate<,,>** oppure traslare con **traslate<,,>**.

- **texture** per definire la superficie: con **pigment** si sceglie il colore.

Gli oggetti geometrici "di base sono: **sphere**, **cylinder**, **box**, **cone**, **torus**, **prism**, **plane**.

Il software POV-Ray è utilizzato dal 2006 dall'**IIS Volta di Pescara** per attività rivolte a studenti particolarmente brillanti ("le eccellenze") con risultati veramente importanti. Tutti i modelli creati (molti sono anche "animati") si possono vedere sul sito: www.itispe.it.

Quest'anno scolastico il "tema" di tale attività sono state le **figure impossibili di Oscar Reutersvard**. E' stata allestita una mostra con alcune delle opere originali (fornite dall'associazione FORMATH), legata alla mostra sono state organizzate esperienze laboratoriali. Una di questa era proprio realizzare in grafica una delle opere in mostra. Particolarmente interessante il progetto realizzato dallo studente Alessandro Lodi, classe IV A Informatica. Lo studente Lodi ha preso spunto, analizzato, studiato le figure di Reutersvard e poi ne ha creata una sua, originale. Quindi: è partito dalla situazione/problema (rappresentare una delle figure), ma è andato oltre: ne ha inventata una sua, ha creato un altro problema! Infine l'ha costruita con POV-RAy e l'ha anche animata.

Vediamo uno stralcio del codice sorgente:

```
  #declare ASSE_Z = sphere{<0,0,0>,200 pigment{Grey} sca-
le<0.001,0.001,0>}
  #declare ASSE_X = object{ASSE_Z rotate 90*y}
  #declare ASSE_Y = object{ASSE_Z rotate 90*x}
  #declare ASSI_XYZ =
  union{
    object{ASSE_Z}
    object{ASSE_X}
    object{ASSE_Y}
  }
  #declare quad =
  difference{
    box{
```

```
    -1,1
    scale<3.4,26,20>
  }
  box{
    -1,1
    scale<6,18,12>
  }
  translate 3*x + -z*3
  pigment{Cyan}
```

Quali "oggetti matematici"? Sfera, parallelepipedo e poi rotazioni, traslazioni.

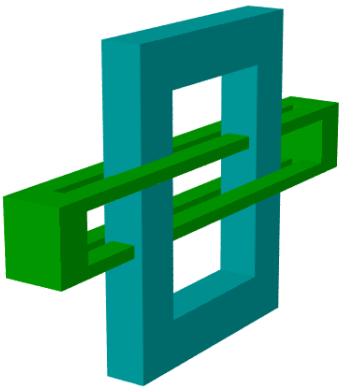
Ancora:

```
union{
  difference{
    object{quad}
    plane{
      y, LY-1
      rotate 24*z
    }
  }
  intersection{
    object{quad}
    plane{
      y, LY-13.67
      rotate 30*z
    }
  }
  difference{
    object{quad}
    plane{
      y, LY-10
      rotate 28*z
    }
    plane{
      z,0
    }
  }
```

```
intersection{
  object{quad}
  plane{
    y, LY-4.6
    rotate 26*z
  }
  plane{
    z,0
  }
```

In questo secondo segmento si possono ricavare le operazioni insiemistiche: difference, union, intersection.

Qual è stato il prodotto finale? Quale oggetto? Eccolo:



Lasciamo al lettore di trovare "l'impossibilità" della figura realizzata da Alessandro Lodi.