Αξιοποίηση Τεχνικών Παράλληλου Προγραμματισμού στο Περιβάλλον Scratch

Δ. Λαδιάς[1], Θ. Καρβουνίδης[2], Α. Λαδιάς[3]

[1]Φοιτητής Τμήματος Πληροφορικής ΕΚΠΑ
ladimitr@gmail.com
[2]Δρ. Εκπαιδευτικός Πληροφορικής
tkarv@unipi.gr
[3]Δρ. Πληροφορικής
ladiastas@gmail.com

Περίληψη

Η εργασία επιχειρεί να δώσει στοιχεία παράλληλου προγραμματισμού, προσαρμοσμένα στο μαθησιακό επίπεδο των μαθητών της υποχρεωτικής εκπαίδευσης μέσα από την παιδαγωγική προσέγγιση του αναδυόμενου γραμματισμού ώστε να αναπτύξουν τις δικές τους ιδέες και νοητικά μοντέλα για το πως μπορούν διεργασίες να εξελίσσονται παράλληλα στο χρόνο. εμπλοκή των μαθητών με τον παράλληλο προγραμματισμό μπορεί να βοηθήσει την ωρίμανση της σκέψης τους προς αυτή την κατεύθυνση. Η εργασία θα πρέπει να ενταχθεί στο πλαίσιο έρευνας για την ανάπτυξη ενός πλαισίου αξιολόγησης του κώδικα Scratch των μαθητών στο εκπαιδευτικό προγραμματιστικό περιβάλλον Scratch, συγκεκριμένα οι διάφορες εκδοχές παράλληλου προγραμματισμού που μπορούν να παρατηρηθούν στο Scratch να κατηγοριοποιηθούν στα επίπεδα της ταξινομίας SOLO.

Λέξεις κλειδιά: Παράλληλος προγραμματισμός, Scratch.

1. Εισαγωγή

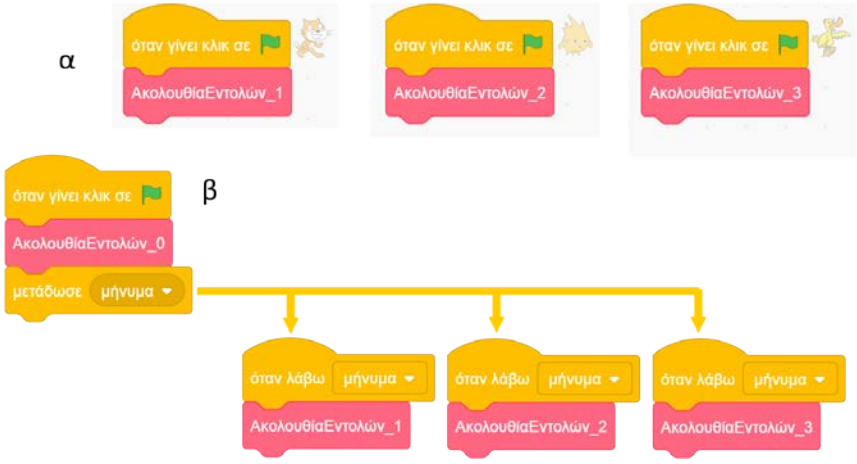
Η παρούσα εργασία επιδιώκει να προσθέσει μια επιπλέον διάσταση, αυτή του παράλληλου προγραμματισμού, στα ήδη προτεινόμενα Προγράμματα Σπουδών για τον Προγραμματισμό Η/Υ με το Scratch για το Δημοτικό Σχολείο (Λαδιάς, Γώγουλος, 2017) και το Γυμνάσιο (Λαδιάς, Καρβουνίδης & Μπέλλου, 2020). Η έννοια "παράλληλα" είναι χωρική και εδώ χρησιμοποιείται ως μεταφορά στο χρόνο, για να περιγράψει τα συγχρόνως εκτελούμενα (concurrent) προγράμματα. Το Scratch είναι μια γλώσσα προγραμματισμού στην οποία είναι δυνατόν τμήματα προγραμμάτων (διεργασίες) να εκτελούνται παράλληλα, για την ακρίβεια ψευδοπαράλληλα αφού η εκτέλεση γίνεται από ένα επεξεργαστή (Schiper, 1989) και αυτό επιτυγχάνεται με εσωτερικό μηχανισμό της γλώσσας που υλοποιείται με κατανομή χρόνου σε έκαστο τμήμα. Για αυτά τα τμήματα προγραμμάτων στο Scratch, στο εξής θα χρησιμοποιείται ο όρος σενάρια, ενώ ως πρόγραμμα θα νοείται το σύνολο των σεναρίων. Στο Scratch τα σενάρια ξεκινούν με εντολές-καπελάκια και

ISBN: 978-960-578-072-2

αντιστοιχούν σε συμβάντα (Εικόνα 1) που ενεργοποιούνται είτε από εξωτερικές ενέργειες (π.χ. αλληλεπίδραση με άνθρωπο ή από εξωτερική συσκευή) είτε από εντολές στο εσωτερικό του προγράμματος όπως οι εντολές αποστολής μηνυμάτων ή δημιουργίας κλώνων.

Εικόνα 1. *Ενδεικτικά είδη σεναρίων που διαχειρίζονται την ανίχνευση συμβάντων προερχόμενα από εξωτερικές ενέργειες όπως η δράση του χρήστη (α1) ή μιας συσκευής π.χ. ρομπότ WeDo ή μικρόφωνο (α2) και από την αποστολή μηνύματος (β1) ή τη δημιουργία κλώνου (β2).*

Στο Scratch υπάρχει η δυνατότητα τα σενάρια να εκτελούνται ταυτόχρονα. Στην Εικόνα 2α εμφανίζεται μια τέτοιου είδους παράλληλη εκτέλεση σεναρίων που παρατηρείται συχνά στους κώδικες μικρών μαθητών που χρησιμεύει για την αρχικοποίηση των θέσεων στη σκηνή των διαφόρων αντικειμένων.

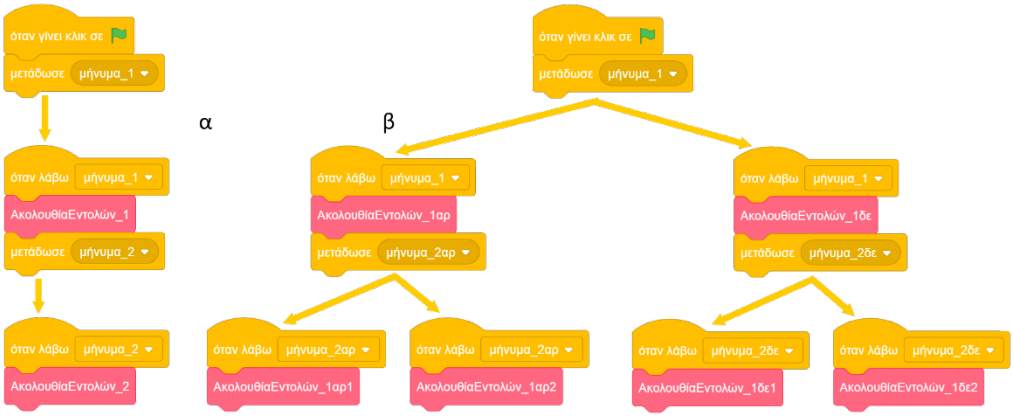


Εικόνα 2. *Τρία σενάρια που: (α) το καθένα ανήκει σε διαφορετικό αντικείμενο, τα οποία εκτελούνται παράλληλα κατά την εκκίνηση του προγράμματος και (β) ενεργοποιούνται από το αρχικό και ξεκινούν ταυτόχρονα την παράλληλη εκτέλεσή τους.*

Μια συχνά παρατηρούμενη διάταξη παράλληλων σεναρίων σε κώδικες μαθητών είναι η αποστολή μηνύματος που το λαμβάνουν ταυτόχρονα πολλά σενάρια-αποδέκτες (Εικόνα 2β). Στο εσωτερικό της ακολουθίας των εντολών ενός σεναρίου

ISBN: 978-960-578-072-2

είναι δυνατόν να εμπεριέχονται και άλλες εντολές «μετάδωσε μήνυμα…» ή «δημιούργησε κλώνο…» με αποτέλεσμα να δημιουργούνται νήματα σεναρίων που μπορούν να είναι γραμμικά ή να σχηματίζουν δενδροειδείς δομές (Εικόνα 3).



Εικόνα 3. *Νήματα σεναρίων με δομή: (α) γραμμική και (β) δενδροειδή.*

2. Παιδαγωγική προσέγγιση της διδασκαλίας του Παράλληλου Προγραμματισμού

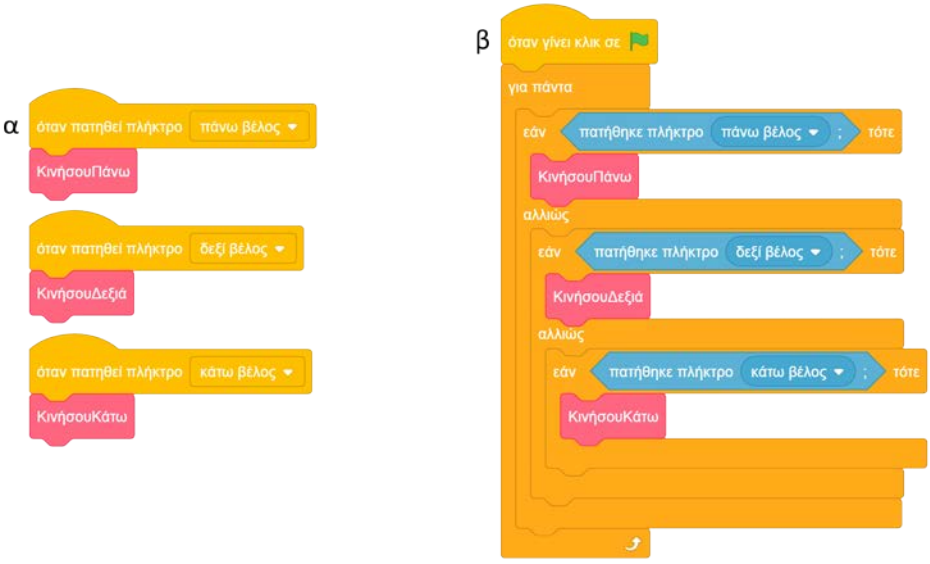
Θα πρέπει να ληφθεί υπόψη ότι η όλη προσπάθεια για την εξοικείωση με τον παράλληλο προγραμματισμό αφενός απευθύνεται σε αρχάριους προγραμματιστές μαθητές της υποχρεωτικής εκπαίδευσης και αφετέρου εντάσσεται σε ένα πλαίσιο εξοικείωσης αυτών και με άλλα προγραμματιστικά στυλ όπως ο προγραμματισμός που καθοδηγείται από γεγονότα, ο προγραμματισμός που βασίζεται σε αντικείμενα, ο αντικειμενοστραφής προγραμματισμός, ο δομημένος προγραμματισμός αλλά και συνδυασμοί τους. Σε αυτές τις πρώιμες ηλικίες η θεωρητική προσέγγιση της διδασκαλίας του προγραμματισμού προτείνεται να είναι αυτή του *αναδυόμενου εγγραμματισμού*. Τα παιδιά -μέσα από μια σπειροειδή προσέγγιση- αρχικά πράττουν, συσσωρεύσουν εμπειρίες και αποκτούν προσλαμβάνουσες παραστάσεις, δίνοντας έμφαση σε διαχρονικές έννοιες του προγραμματισμού και όχι σε εξειδικευμένες δυνατότητες εφήμερων γλωσσών προγραμματισμού. Αργότερα και καθώς ωριμάζουν στο μυαλό τους οι ιδέες, έρχεται η θεωρία που συστηματοποιεί τις εμπειρίες που έχουν αποκτηθεί μέσα από την πράξη. Έτσι οι μαθητές εξοικειώνονται με έννοιες υψηλού επιπέδου χωρίς να χρησιμοποιούνται ορισμοί, χωρίς να διδάσκονται θεωρητικά οι δομές και οι έννοιες, τις οποίες ξεκαθαρίζουν σταδιακά και με τη συστηματική χρήση τους (Λαδιάς, 2011).

Σε αυτό το πλαίσιο διερευνώνται οι δυνατότητες ψευδοπαράλληλου προγραμματισμού στο Scratch και γι’ αυτό το λόγο στο εξής δεν έχει νόημα να γίνεται διάκριση μεταξύ των όρων ταυτόχρονα (concurrent) εκτελούμενα σενάρια

ISBN: 978-960-578-072-2

(στα οποία ορίζονται δράσεις που μπορούν να εκτελούνται ταυτόχρονα), παράλληλα σενάρια (που είναι ταυτόχρονα εκτελούμενα σενάρια που εκτελούνται σε παράλληλους επεξεργαστές) και κατανεμημένα (distributed) σενάρια (που είναι παράλληλα σενάρια για εκτέλεση σε δίκτυο αυτόνομων επεξεργαστών που δεν διαμοιράζονται την κύρια μνήμη) (Bal, Steiner & Tanenbaum, 1989).

Σύμφωνα με τον Kahn (1996) ο παράλληλος προγραμματισμός είναι πιο κοντά στο φυσικό τρόπο σκέψης των μαθητών, αφού στον πραγματικό κόσμο ανεξάρτητες οντότητες λειτουργούν ταυτόχρονα και επικοινωνούν μεταξύ τους, ενώ υπάρχουν προτάσεις για την υιοθέτηση ταυτόχρονων γλωσσών για την πρώτη επαφή των μαθητών με τις γλώσσες προγραμματισμού.

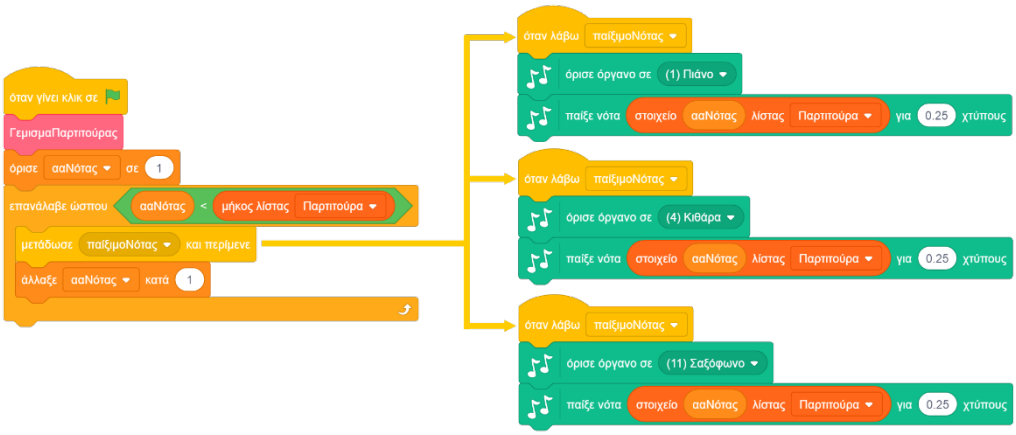


Εικόνα 4.*(α) Τα τρία σενάρια που τρέχουν ταυτόχρονα σε ένα πρόγραμμα με προσέγγιση παράλληλου προγραμματισμού (β) το ισοδύναμο σειριακό πρόγραμμα.*

Με αυτή την άποψη συμφωνούν και τα ευρήματα προς δημοσίευση έρευνας που υποστηρίζουν ότι ο τρόπος που προγραμματίζουν οι αρχάριοι μαθητές είναι πλησιέστερα στο συνδυασμό προγραμματισμού που βασίζεται σε γεγονότα συνδυαζόμενου με τον παράλληλο προγραμματισμό έναντι του σειριακού / δομημένου προγραμματισμού (Εικόνα 4).

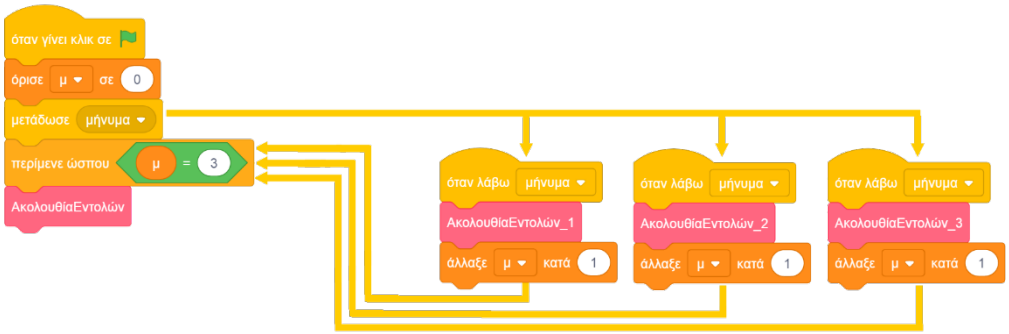
Η δυνατότητα να εκτελούνται ταυτόχρονα τα σενάρια δεν αποτελεί μια εναλλακτική λύση έναντι του σειριακού προγραμματισμού αλλά για ορισμένα είδη προβλημάτων είναι αναγκαιότητα. Ένα τέτοιο παράδειγμα είναι αυτό της εικόνας 5 κατά το οποίο κάθε νότα μιας παρτιτούρας παίζεται ταυτόχρονα από περισσότερα του ενός μουσικά όργανα.

ISBN: 978-960-578-072-2



Εικόνα 5. *(α) Πρόγραμμα στο οποίο περισσότερα του ενός μουσικά όργανα παίζουν ταυτόχρονα την ίδια νότα μιας παρτιτούρας (σε λίστα) εκμεταλλευόμενο την παράλληλη εκτέλεση των αντίστοιχων σεναρίων.*

Η δυνατότητα παράλληλης εκτέλεσης των σεναρίων στο Scratch προϋποθέτει την επικοινωνία (για τον συγχρονισμό) μεταξύ των σεναρίων με τη χρήση αντίστοιχων μηχανισμών (Εικόνα 6), οι οποίοι αν και είναι διαθέσιμοι στο Scratch, δεν είναι εμφανείς στον προγραμματιστή, με αποτέλεσμα αυτός να πρέπει να βασίζεται στις δικές του νοητικές αναπαραστάσεις (Nikolos & Komis, 2015).

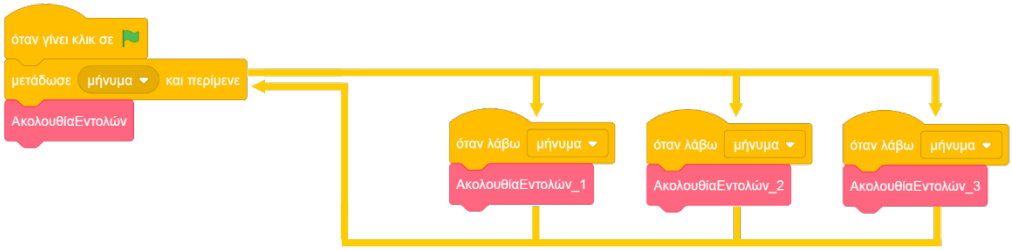


Εικόνα 6. *Μηχανισμός συγχρονισμού μεταξύ του αρχικού σεναρίου και των τριών παράλληλων σεναρίων. Το αρχικό σενάριο θα περιμένει να ολοκληρωθούν και τα τρία παράλληλα σενάρια πριν εκτελέσει την «ΑκολουθίαΕντολών»*

Αυτό έχει ως αποτέλεσμα ο συγχρονισμός και η επικοινωνία σε προγραμματιστικά περιβάλλοντα με δυνατότητες παράλληλου προγραμματισμού, να φαίνεται ότι είναι ένα πολύπλοκο έργο (Ben-Ari, 2006). Όμως η πίστη ότι η ταυτόχρονη εκτέλεση διεργασιών είναι πολύ δύσκολη για τον μέσο προγραμματιστή δεν θεμελιώνεται Bustard (1990). Σε αυτό συμφωνεί και ο Rifkin (1994) που δηλώνει πως η διδασκαλία των πολύπλοκων αυτών εννοιών δεν είναι κατ' ανάγκη δύσκολη.

ISBN: 978-960-578-072-2

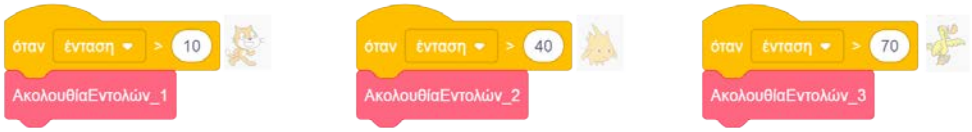
Σύμφωνα με την Καψιμάλη (2010), η διδασκαλία του παράλληλου προγραμματισμού εξυπηρετεί τους ευρύτερους στόχους της εκπαίδευσης στην επιστήμη υπολογιστών, επεκτείνοντας κάποιες από τις βασικές αρχές της (Ben-Ari & Kolikant, 1998). Στην εικόνα 7 φαίνεται ισοδύναμος κώδικας με αυτόν του σχήματος 6 που υλοποιείται απλούστερα χρησιμοποιώντας τις ενσωματωμένες δυνατότητες συγχρονισμού του Scratch και είναι ένα παράδειγμα αυτής της σχετικής ευκολίας με την οποία γίνονται κατανοητές αυτές οι πολύπλοκες έννοιες.



Εικόνα 7. *Η εντολή «μετάδωσε ... και περίμενε» εμπεριέχει ένα μηχανισμό ελέγχου του τερματισμού όλων των παράλληλων σεναρίων που πυροδότησε.*

3. Στοιχεία τοπολογίας παράλληλων σεναρίων

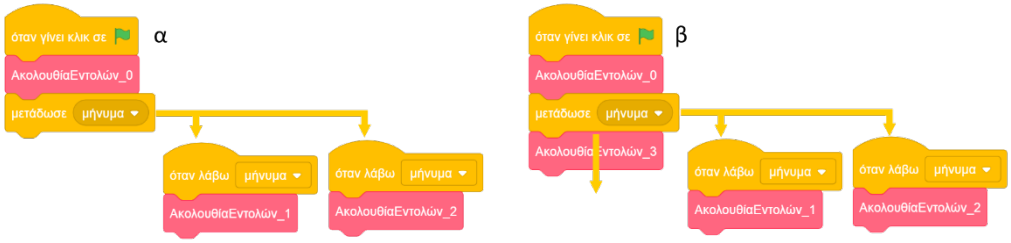
Συνήθως ένα πρόγραμμα είτε αρχίζει σειριακά από ένα σενάριο και στη συνέχεια δημιουργεί περισσότερα παράλληλα νήματα (Εικόνες 2β, 3β και 5), είτε ξεκινά εξ αρχής με παράλληλα νήματα όπως στις εικόνες 2α και 4α. Και στις δύο περιπτώσεις τα παράλληλα νήματα μπορεί είτε να ξεκινούν να εκτελούνται ταυτόχρονα (Εικόνες 2α και 2β), είτε να ενεργοποιούνται ξεχωριστά ανάλογα με κάποιο συμβάν (Εικόνες 4α και 8).



Εικόνα 8. *Τρία σενάρια που εκτελούνται παράλληλα αλλά δεν ξεκινούν ταυτόχρονα. (α) Κανένα σενάριο δεν εκτελείται αν η ένταση του μικροφώνου είναι μικρότερη του 10, (β) το σενάριο Α εκτελείται μόνο του αν η ένταση είναι στο διάστημα (10 – 40], (γ) τα σενάρια Α και Β εκτελούνται ταυτόχρονα αν η ένταση είναι στο διάστημα (40 – 70] και (δ) και τα τρία σενάρια εκτελούνται ταυτόχρονα όταν η ένταση είναι μεγαλύτερη του 70.Τα σενάρια θα μπορούσαν να είναι και σε ένα μόνο αντικείμενο.*

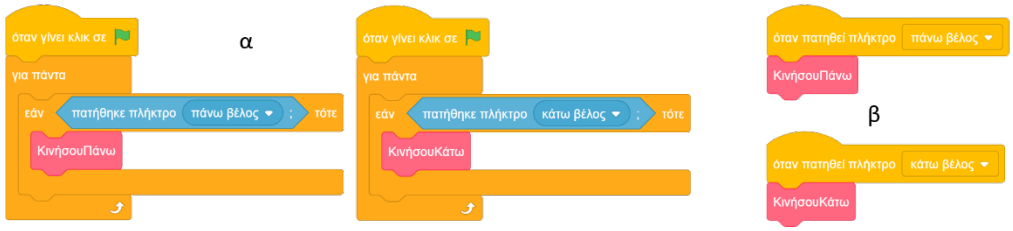
Αξίζει να σημειωθεί η διαφορά των δύο προγραμμάτων της Εικόνας 9. Στο σχήμα 9α το αρχικό σενάριο τερματίζεται μετά την εντολή «μετάδωσε…» που δημιουργεί τα παράλληλα σενάρια ενώ στο πρόγραμμα του σχήματος 9β και το αρχικό σενάριο συνεχίζει να εκτελείται παράλληλα με τα νέα παράλληλα σενάρια που δημιούργησε.

ISBN: 978-960-578-072-2



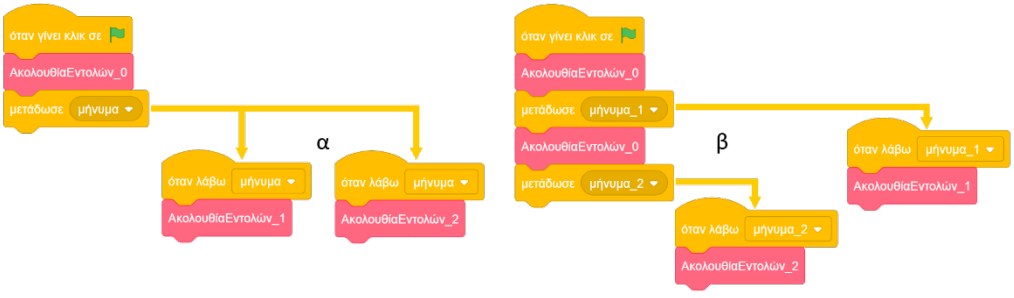
Εικόνα 9. *Το πατρικό σενάριο μετά τη δημιουργία των παράλληλων σεναρίων (α) τερματίζεται ενώ στο (β) συνεχίζει να «τρέχει» παράλληλα με τα νέα σενάρια.*

Οι καταστάσεις στις οποίες μπορεί να βρεθεί κάθε φορά ένα σενάριο είναι: (α) η *ενεργή κατάσταση*, όπου το σενάριο είτε τρέχει / εκτελείται (Εικόνα 10α) είτε είναι σε επαγρύπνηση συμβάντος και έτοιμο προς εκτέλεση (Εικόνα 10β), (β) η *κατάσταση αδράνειας*, όπου το σενάριο έχει αδρανοποιηθεί περιμένοντας κάτι να ολοκληρωθεί για να το επαναδραστηριοποιήσει όπως τα τρία σενάρια στο δεξιό τμήμα της εικόνας 6 και (γ) η *κατάσταση λήξης*, όπου η εκτέλεσή του σεναρίου έχει τελειώσει όπως το πατρικό σενάριο της εικόνας 9α.



Εικόνα 10. *Παράλληλα σενάρια σε ενεργή κατάσταση: (α) Τα σενάρια εκτελούνται και (β) τα σενάρια είναι σε επαγρύπνηση συμβάντος και έτοιμα προς εκτέλεση.*

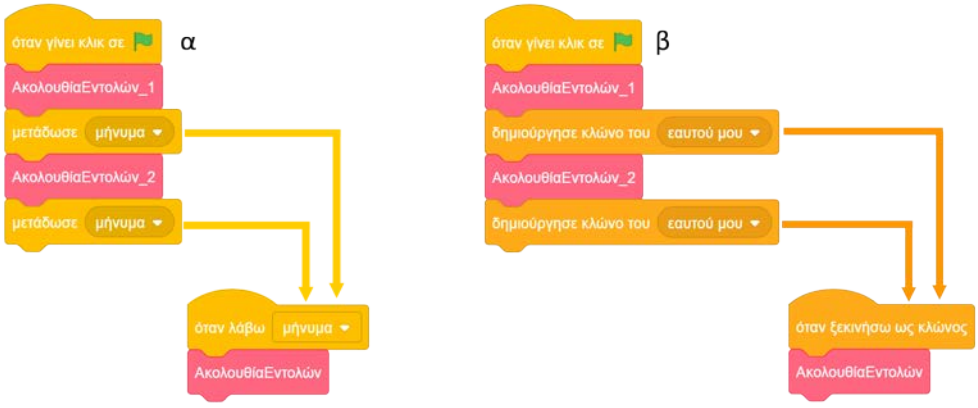
Επίσης τονίζεται ότι η δημιουργία παράλληλων σεναρίων μπορεί να γίνει είτε ταυτόχρονα για δίδυμα σενάρια (Εικόνα 11α), είτε διαδοχικά (Εικόνα 11β), ενώ ενδεχομένως μπορεί να υπάρξει και συνδυασμός αυτών.



Εικόνα 11: *Τα νέα σενάρια (που εκτελούνται παράλληλα) στο πρόγραμμα (α) ενεργοποιούνται ταυτόχρονα ενώ στο (β) ενεργοποιούνται διαδοχικά.*

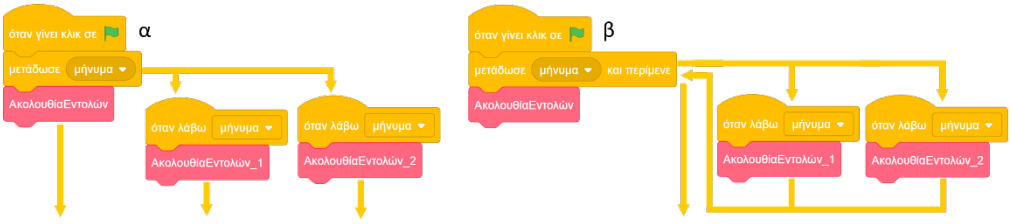
ISBN: 978-960-578-072-2

Παραλληλία μπορεί να υπάρξει είτε όταν στο πρόγραμμα συνυπάρχουν δύο (ή περισσότερα) σενάρια που καλούνται προς εκτέλεση (Εικόνα 11) είτε κατά την εκτέλεση όταν το ίδιο σενάριο ενεργοποιείται πολλαπλές φορές σε διαδοχικές χρονικές στιγμές (Εικόνα 12).



Εικόνα 12: *Παραλληλία μέσω της διαδοχικής χρονικά εκτέλεσης του ιδίου σεναρίου, υλοποιούμενου με (α) μηνύματα και (β) κλώνους.*

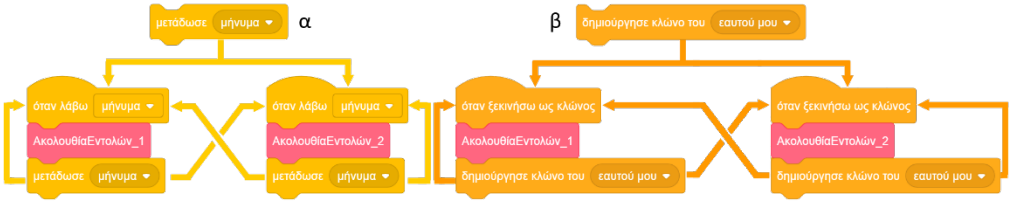
Σύμφωνα με τον Bustard (1990) -ο οποίος αναφέρεται σε ταυτόχρονες διεργασίες- υπάρχουν δύο κύρια μοντέλα εκτέλεσης των παραλλήλων σεναρίων: (α) Τα παράλληλα σενάρια, όταν ενεργοποιηθούν εκτελούνται ανεξάρτητα από το σενάριο που πυροδότησε την εκτέλεσή τους (Εικόνα 13α) και (β) τα παράλληλα σενάρια δημιουργούν ροές νημάτων που διακλαδώνονται (Εικόνα 3β) και τα οποία όταν ολοκληρωθούν η ροή τους συμβάλλει σε ένα και μοναδικό σενάριο ξανά (Εικόνα 13β). Τρόποι με τους οποίους μπορεί να ελέγχεται η ολοκλήρωση της εκτέλεσης των παράλληλων σεναρίων έχουν αναφερθεί στις εικόνες 6 και 7.



Εικόνα 13: *Τα δύο κύρια μοντέλα εκτέλεσης παράλληλων σεναρίων όπου τα δημιουργούμενα παράλληλα σενάρια: (α) αυτονομούνται από το σενάριο που τα δημιούργησε και (β) αφού ολοκληρώσουν τα καθήκοντά τους επανέρχονται στη ροή του αρχικού σεναρίου.*

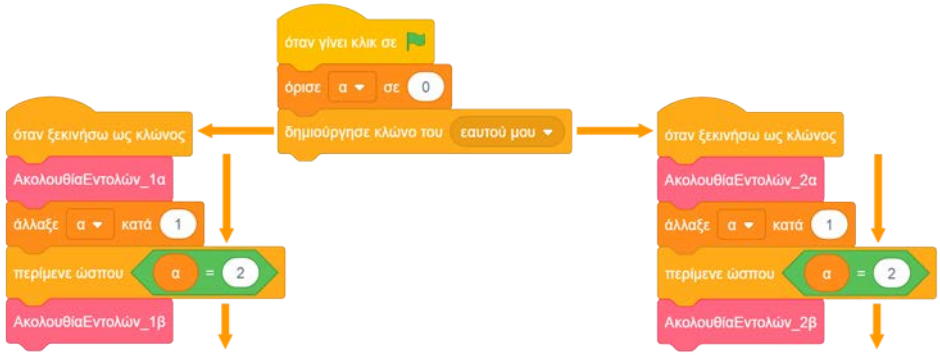
ISBN: 978-960-578-072-2

Ένα σειριακό πρόγραμμα όταν εκτελεστεί πολλές φορές με τα ίδια δεδομένα εισόδου θα δώσει τα ίδια αποτελέσματα, δηλαδή υπάρχει μια λογική σχέση αιτίας-αποτελέσματος (αιτιοκρατία), ενώ ένα παράλληλο πρόγραμμα μπορεί να παρουσιάζει *μη-αιτιοκρατική συμπεριφορά*. Αυτό οφείλεται γιατί όταν ένα παράλληλο πρόγραμμα εκτελείται πολλές φορές με τα ίδια δεδομένα εισόδου, αυτό μπορεί να ακολουθεί κάθε φορά διαφορετικά μονοπάτια εκτέλεσης (Bustard, 1990) που δεν είναι δυνατόν να προβλεφθούν. Παράδειγμα μη-αιτιοκρατικής συμπεριφοράς είναι το πρόγραμμα του σχήματος 14 στο οποίο κατά την εκτέλεση η ροή του προγράμματος είναι ανεξέλεγκτη λόγω των υπαρχόντων βρόχων που δημιουργούνται.



Σχήμα 14: *Παράλληλα προγράμματα που παρουσιάζουν κατά την εκτέλεσή τους μη-αιτιοκρατική συμπεριφορά υλοποιούμενα (α) με μηνύματα και (β) με κλώνους.*

Για να ελεγχθεί η μη-αιτιοκρατική συμπεριφορά στον παράλληλο προγραμματισμό αναπτύχθηκαν προγραμματιστικές τεχνικές που εξασφαλίζουν το συγχρονισμό της ροής των παράλληλων νημάτων σεναρίων. Συνήθως τα παράλληλα νήματα σεναρίων ανταγωνίζονται για την πρόσβασή τους **σε κρίσιμες περιοχές του κώδικα** στις οποίες γίνεται η διαχείριση διαμοιραζόμενων πόρων (π.χ. λίστες δεδομένων ή φυσικές συσκευές). Έτσι προκύπτει η ανάγκη να **συγχρονίζονται** μεταξύ τους για να αποφύγουν τις **συγκρούσεις**.



Σχήμα 15: *Πρόγραμμα στο οποίο δύο παράλληλα σενάρια συγχρονίζονται κατά την εκτέλεσή τους ώστε να συνεχίσουν ταυτόχρονα.*

Παράδειγμα συγχρονισμού μεταξύ δύο παράλληλων σεναρίων είναι αυτό της εικόνας 15 στο οποίο ο κλώνος που θα φτάσει πρώτος στην εντολή «περίμενε ώσπου…» θα τεθεί σε κατάσταση αδράνειας έως ότου να φτάσει και ο δεύτερος κλώνος στο ίδιο

ISBN: 978-960-578-072-2

σημείο, για να συνεχίσουν από αυτό το σημείο την εκτέλεσή τους ταυτόχρονα. Παρόμοιο παράδειγμα συγχρονισμού της ροής ενός παράλληλου προγράμματος είναι και αυτό της εικόνας 6.

Συνήθως ο τερματισμός ενός σεναρίου στο Scratch δηλώνεται δια της απουσίας άλλων εντολών. Όμως υπάρχει η εντολή «σταμάτησε…» του Scratch που αν και χρησιμοποιείται σπάνια από τους μαθητές είναι χρήσιμη αφενός στα πλαίσια του savoir vivre για ευανάγνωστο προγραμματισμό, αφετέρου καλύπτει μερικές φορές συγκεκριμένες αλγοριθμικές ανάγκες. Έτσι η εντολή «σταμάτησε…» μπορεί να χρησιμοποιηθεί για να καθορίσει τους διάφορους τρόπους που θα τελειώσει ένα παράλληλο πρόγραμμα. Στο σχήμα 16 φαίνονται δύο τρόποι τερματισμού των παράλληλων σεναρίων με τη χρήση δύο εκδοχών της εντολής «σταμάτησε…» όπου το πρόγραμμα τερματίζεται: (α) όταν κάποιο σενάριο τελειώσει πρώτο διακόπτοντας τότε την εκτέλεση των υπολοίπων σεναρίων και (β) όταν τερματίσει και το τελευταίο παράλληλο σενάριο δηλ. όταν τελειώσουν όλα τα σενάρια. Το πρόγραμμα αυτό (Σχήμα 16β) είναι ισοδύναμο με αυτό του σχήματος 2α όπου ο τερματισμός των σεναρίων δηλώνεται δια της απουσίας άλλων εντολών.

Σχήμα 16. *Τερματισμός παράλληλων σεναρίων με το πρόγραμμα να ολοκληρώνεται όταν (α) τερματίσει το πρώτο σενάριο και (β) τελειώσουν όλα τα σενάρια.*

3. Επίλογος

Στα προηγούμενα δόθηκαν πολλαπλές προσεγγίσεις των στοιχείων του παράλληλου προγραμματισμού, προσαρμοσμένων στο αντίστοιχο ηλικιακό και μαθησιακό επίπεδο των μαθητών και αποφεύγοντας εξεζητημένα θέματα (π.χ. υλοποίηση διεργασιών-νημάτων, σηματωροί κ.λπ.) ώστε να είναι συμβατό με τα προτεινόμενα προγράμματα σπουδών της υποχρεωτικής εκπαίδευσης. Υπενθυμίζεται ότι η παιδαγωγική προσέγγιση είναι αυτή του αναδυόμενου γραμματισμού που στοχεύει κυρίως στην απόκτηση βιωματικών εμπειριών από τους μαθητές και όχι στην αποστήθιση επιστημονικών ορισμών. Στο υλικό που χρησιμοποιήθηκε στην παρούσα εργασία επιδιώχθηκε να δοθούν στους μαθητές ερεθίσματα για να αναπτύξουν τις δικές τους ιδέες και νοητικά μοντέλα για το πως μπορούν διεργασίες να εξελίσσονται παράλληλα στο χρόνο. Ενδεχομένως μια δυσκολία που θα παρουσιαστεί είναι η ανωριμότητα όσον αφορά την αντίληψη του χρόνου ιδιαίτερα στις πρώιμες ηλικίες της υποχρεωτικής εκπαίδευσης. Όμως ταυτόχρονα η εμπλοκή των μαθητών με τον παράλληλο προγραμματισμό μπορεί να βοηθήσει την ωρίμανση της σκέψης τους προς αυτή την κατεύθυνση.

ISBN: 978-960-578-072-2

Το υλικό το σχετικό με τον παράλληλο προγραμματισμό που αναπτύχθηκε στην παρούσα εργασία θα πρέπει -ενδεχομένως μέσω διπλωματικών εργασιών- να αξιολογηθεί ως προς την παιδαγωγική του διάσταση και τον τρόπο εφαρμογής του στην τάξη, με την αναζήτηση στοιχείων παράλληλου προγραμματισμού σε αποθετήρια κωδίκων Scratch όπως αυτά του Πανελλήνιου Διαγωνισμού Εκπαιδευτικής Ρομποτικής του WRO-Hellas ή του Πανελλήνιου Διαγωνισμού Scratch-Game.
Επίσης η διάσταση του παράλληλου προγραμματισμού θα πρέπει να ενταχθεί στο πλαίσιο της έρευνας για την ανάπτυξη ενός πλαισίου αξιολόγησης του κώδικα Scratch των μαθητών στο εκπαιδευτικό προγραμματιστικό περιβάλλον Scratch (Karvounidis, Argyriou, Ladias, & Douligeris, 2017), η οποία εξελίσσεται σε δύο άξονες που εξετάζουν την ανατομία και τη λειτουργικότητα του κώδικα. Συγκεκριμένα θα πρέπει να ενταχθεί στον άξονα της λειτουργικότητας επιδιώκοντας να κατηγοριοποιηθούν στα επίπεδα της ταξινομίας SOLO οι διάφορες εκδοχές παράλληλου προγραμματισμού που μπορούν να παρατηρηθούν στο Scratch.

Αναφορές

Bal, H. E., Steiner, J. G., and Tanenbaum, A. S. Programming Languages for Distributed Computing Systems. *ACM Computing Surveys 21, 3 (Sept. 1989), 261-322.*

Ben-Ari, M. & Kolikand, Y. B.-D. (1998). Thinking Parallel: The process of Learning Concurrency. ITiCSE, 13-16 Cracow.

Ben-Ari, M. (2006). *Principles of Concurrent and Distributed Programming*. Essex: Addison-Wesley.

Bustard, W., D. (1990). Concepts of Concurrent Programming. *Software Engineering Institute / Carnegie Mellon University (SEI-CM-24)*.

Harel, D. “On Visual Formalisms.” Comm. *ACM 31, 5 (May 1988), 514-530.*

Kahn, K. (1996). Drawing on napkins, video-game animation, and other ways to programma computers. Communications of the ACM, 39(8), 49-59.

Karvounidis, Th., Argyriou, I., Ladias, An., Douligeris, Chr. (2017). A Design and Evaluation Framework for Visual Programming Codes. *The IEEE Global Engineering Education Conference(EDUCON2017). Athens.*

Nikolos, D., & Komis, V. (2015). Synchronization in Scratch: A Case Study with Education Science Students. *Jl. of Computers in Mathematics and Science Teaching (2015) 34(2), 223-241*.

ISBN: 978-960-578-072-2

Rifkin, A. (1994). Teaching Parallel Programming and Software Engineering Concepts to High School Students. *Proceedings of the SIGCSE technical symposium on Computer Science Education, pp.26-30. Phoenix: ACM*

Schiper, A. *Concurrent Programming*. Halsted Press; John Wiley & Sons Inc., 1989.

Καψιμάλη, Β., (2010). Τεχνολογικά υποστηριζόμενη διδακτική της πληροφορικής με χρήση του εργαλείου Scratch. *Διπλωματική εργασία. Πειραιάς*.

Λαδιάς, Α., (2011). Ο προγραμματισμός Η/Υ στο νέο Π.Σ. της υποχρεωτικής εκπαίδευσης στο πλαίσιο του μαθήματος για τον Πληροφορικό Γραμματισμό. *3rd CIE– Η Πληροφορική στην Εκπαίδευση. Πανεπιστήμιο Πειραιά, Πειραιάς 2011.*

Λαδιάς, Α., Γώγουλος, Γ. (2017). Ο προγραμματισμός υπολογιστικών συσκευών ως κύριος άξονας ενός Προγράμματος Σπουδών στην Πληροφορική. *Έρκυνα, 13, 158-168.*

Λαδιάς, Α., Καρβουνίδης, Θ., Μπέλλου, Ι. (2020). Πρόταση Προγράμματος Σπουδών για τον προγραμματισμό υπολογιστικών συσκευών στα πλαίσια του STEM. *Έρκυνα, 19, 69-84.*

Abstract

The present work attempts to present elements of parallel programming, adapted to the learning level of primary education students. This attempt is being carried out with the means of the pedagogical approach of the emerging literacy, aiming to develop their own ideas and mental models of how processes can evolve over time. The involvement of students with parallel programming can help them mature their thinking towards this direction. This work should be included in the research to develop a Scratch code evaluation framework for students in the Scratch educational programming environment, namely the various versions of parallel programming that can be observed in Scratch to be categorized at the SOLO taxonomy levels.

Keywords: concurrent / parallel programming, Scratch.

ISBN: 978-960-578-072-2